POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Multi-modal Communication Interactions Between Socially Assistive Robot and People with Neurodevelopmental Disorders

Master of Science in Computer Science and Engineering

Author: Corrado Pacelli and Tharushi Kinkini De Silva Pallimulla Hewa Geeganage

Advisor: Prof. Franca Garzotto

Co-advisor: Micol Spitale

Academic year: 2020-2021

1. Introduction

1.1. Motivation and Goal

Communication, *the human connection*, is the successful conveying or sharing of ideas and feelings indeed it plays a vital role in human life. It could be very difficult for people with Neurodevelopmental Disorders to communicate with others. NDD ranges from 0.4% to 3% of the worldwide population across countries and are characterized by early-onset deficits of variable severity in personal, social, academic, or occupational functioning. These latter often limits them in expressing their ideas, feelings, opinions and in relating to others. It is often taken for granted that human communicates only in a certain way without taking into account the different characteristics that each one has. So why not facilitate communication, in particular using a robot, easing different interaction modalities thus it is the user himself who decides which is the mode in which he feels most comfortable or he likes best. In order to achieve this we rely on assistive technology which has proved to have great potential in healthcare settings and as mental health interventions for children. Robots have been shown to generate a number of social and behavioral benefits in children with ASD including heightened engagement, increased attention, and decreased social anxiety [4]. Our goal is to investigate, among the several interaction modalities, which are the most suitable for this target and from a technological point of view to realize this communication. We would like to improve the critical aspects of children with NDD, in particular autism, through an innovative and positive reinforcement activity that combine Applied behavioural analysis (ABA) therapy with Socially Assistive Robot (SAR).

1.2. Research questions

Our research questions wants to investigate on:

- what is the most suitable interaction approach to communicate for people with NDD while interacting with a robot.
- what is an efficient way to enable the implementation of different communication modalities for adapting for different individual needs.

1.3. Research approach

As means to have a better understanding the existing questions, we settled for exploratory research. We have emphasised qualitative research

methods with the objective to find meanings, observe the target audience behaviour, opinions and preferences. First of all we have created an innovative and positive reinforcement activity with different interaction paradigms: voice, cards and buttons. We came up with a short story on recycling in order to reinforce or teach basic concepts and social skills. After the implementation of the latter, we organized two focus group and three observations with people with NDD. Regarding the implementation we needed a framework that gave us the possibility to:

1. easily add components for the different interaction methods
2. face the challenge of real time computing

The framework that we used is HARMONI. HARMONI is a comprehensive tool containing components and capabilities that developers can use in order to get a social assertive up and running on a robot. Since we also needed a high degree of flexibility in the choice of the execution path to be executed at run time, we decided to enchant HARMONI by empowering it with behavioural trees [3]. To integrate behaviour tree in HARMONI we used PyTree [2], a python library that implements all the components present in the theory of behaviour tree.

1.4. Research contribution

We can identify two research contribution:

- design and method: we have designed an immersive activity that combines both Applied Behavioural Analysis (ABA) therapy with Socially Assistive Robot (SAR) giving the possibility to interact with different paradigms that are verbal, cards and buttons. In particular, given a specific target, we tried through an exploratory research to identify what is the most suitable communication approach, why it is preferred, what is the background of the people concerned and so on.
- implementation: we tried to figure out how to facilitate these interaction paradigms from development point of view. We have implemented inside an open source, Robot-Agnostic, modular and composable framework that is HARMONI, behaviours trees. We have integrated python library for trees and it gives several advantages like simplicity, scalability,modularity, dynamism and priority handling. Moreover we have add a new module in HARMONI that is able to perform image recognition using ImageAI python library.

2. Design

Our activity takes inspiration on social stories and it foresees an interaction between robot and child. In particular we built a little story on recycling. Even if the story is fixed, it can slightly change based on the answers/choices of the child, so we designed some different path. The focus of the activity is on the way in which the child interact and express his/her knowledge in daily routine activities. Moreover we want to educate and raise awareness of the child so if he/she answers 'in a wrong way' at one or more questions we help him/her in understand the correct answer. The main characteristics of the designed solution is the possibility to have three different interaction paradigms: verbal, cards and buttons (Figure 1).

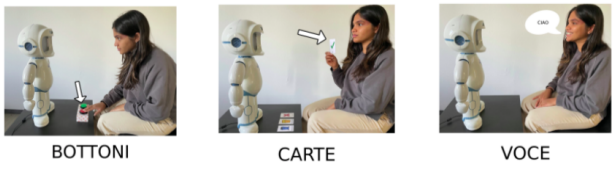


Figure 1: Example of the three interaction paradigms: buttons, cards and voice

2.1. Human Robot Interaction

The interaction between robot and child is thought to be very friendly, we would like that the child feels calm and at ease as he is interacting with a friend. The robot is not a judge, it is a companion inside the story that is interested on child's improvement and behaviour. The robot aside from dialogue, pays close attention, thanks to its incorporated camera, that the child falls within its field of vision. So we have used QTrobot for our activity implementation.

2.2. Activity designed

Our recycling story is focused on paper, plastic and glass. In order to have a more realistic session we have created a story that is similar to a typical morning routine in which they are going to have breakfast and then clear the ta-

ble and throw the garbage in the correct bin. Everything is accompanied by the projection on the screen or wall to make an immersive story. The whole session can be divided into two correlated parts: main activity, background. The main activity part is the one containing the story with its teachings. The story has an introduction part where the robot teaches what is the recycling with some examples and then asks the child where to throw some rubbish. Then there is an interlude where they have to go out to buy milk; this part is created to see the responsiveness and how the child behaves in some circumstances. In our activity we thought to show a cloudy and rainy sky and ask the child if he thinks that they should bring an umbrella with them while leaving home. The story ends with the child and robot going home to have finally breakfast. This latter takes up the issue of the recycling and asks again the child, to reinforce the concepts, where to throw the garbage. The background is the component designed to handle some unexpected action of the child during the main activity. Monitoring measures are done in a precise order: we first perform the visual checking for which we have given the highest priority and then we continue with the interaction one if the previous one was successfully.

- **visual checking**: is responsible for seeing if the child falls within the robot's field of view. If the child is not there the robot will ask the child where is he and to come in front of it.
- **interaction checking**: checks if the child has no interaction for more than two times and then asks him if there's something wrong.

Every time a background checking is activated because either the child is not there or he is not interacting, the main activity is paused and the above mentioned actions are done. If the problem is solved, the main activity is resumed otherwise we call the therapist and stop the whole session.

3. Implementation

So as to achieve our goal we used HARMONI framework [3]. At the beginning of our thesis work this framework was meant to work in a sequential fashion but this paradigm didn't fit our needs properly, we needed to run different modules based on priority order and on dynamic conditions; so our first step was adding behaviour trees in HARMONI environment. Among the several packages that implement behaviour trees we decided to use PyTree [2]. The choice of PyTree was made for many reasons by one of the most important was the fact that it is written in python like the HARMONI framework. Once added PyTree in HARMONI we proceeded with the design of the activity and with its implementation. Activity in PyTree are implemented as behavioural tree so, to create our behavioural tree, we split our work in two part:

- Design and implementation of the *body* of the tree.
- Implementation of all the *leaves* (Behaviours) thought in the design.

The steps that bring to the construction of the body of the tree are quite a lot. We went through three phases such as design, implementation and test. We reiterated these three phases two times: the first one was for the creation of the skeleton without consider leaves, the second one was done after the creation of the leaves and it was an adjustment phase. For all these stages we decided to apply a "top-down" approach so we define first a big picture of the tree and then we went in details. We created first the subtree related to the main activity and then we thought about the backgrounds subtrees. Regarding backgrounds, they take care of some unexpected situations that might happen during the activity and, when such situations occur, they are in charge of running special procedures. We have two backgrounds modules: the first one is called *Visual* and the second one *Interaction*. The first one come in play when the robot can't detect a person in front of it while the second one is played when the user doesn't interact for a specific period of time. Figure 2 shows the subtree that controls the execution of main activity and backgrounds. Children of the Session node are arranged in a priority order, so the first child is the most important one and so on. In Figure 2 the first children is the subtree *Therapist* that is in running all the times the activity has to be interrupter and the therapist has to come in play. Once the design of the body was finished we went on to create leaves of the tree. Each leaf in PyTree is a *Behaviour*, so in order to add a leaf to the tree it has to extend the Behaviour

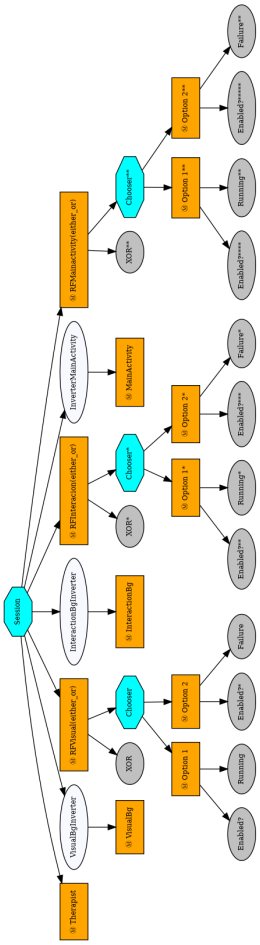


Figure 2: This figure shows the root of the activity controller. The *Session* selector controls the execution flow of the two backgrounds, of the main activity and the therapist intervention. It is in charge of change the execution between modules accordingly to the answers and actions of the user.

class. We created 16 different leaves that can be grouped in two main categories: *request* leaves and *on* leaves. Leaves are implemented as clients of HARMONI servers so they all make request to services. The difference in the categories is indeed a difference in the type of request that clients (leaves) make. *Request* leaves perform a single request to the server and then they wait for the return value. *On* leaves perform a single request that ask to the server to *start* functioning. In the second group we can find all the modules that are suppose to be sensor like microphone and camera. Among the most complex leaves, we report here: Amazon Lex, Face Detection, Card Detection and Buttons. The first leaf makes use of the well-known service of Amazon Web Service (AWS) to create conversational agents. Indeed we created a bot that took care of the conversation between robot and user. The two leaves for the detection make use of YoloV3 neural network. We used ImageAI [1] to create our neural network. Face Detection leaf makes use of the plain YoloV3 neural network filtering only for person. Regarding Card Detection we create our dataset of cards (Figure 3) and we re-trained the YoloV3. In Figure 4 there is an example of output of our neural network for card detection. Buttons have not been implemented

Figure 3: Deck of cards that we used for the creation of the dataset and for the training of the Neural Network. The cards are also the ones used during the activity

Figure 4: Output of one frame of the NN that recognize cards with a probability of 97.93% .

as client-server, they are attached to the robot and the leaf that controls them is just listening on the serial where buttons publish their results. We created physical buttons by using Wemos d1 chipset.

4. Exploratory Research

We proceeded by doing an exploratory research and, among the different exploratory research methods, we have settled for two focus groups and three observations. Exploratory Research is defined as a research used to investigate a problem which is not clearly defined. It is conducted to have a better understanding of the existing problem, but will not provide conclusive results. The focus group comprised each day six, seven participants with NDD but having a fairly high level cognitive skills respect than others with same disorders. During these sessions we had the possibility to collect ideas, opinions, suggestions related our work and most of all on what they would have preferred both on interaction paradigm and activities. Whereas the observations days were based on individual testing of our implemented activity with QTrobot. During the observation we have collected information on how they prefer to interact with the robot (verbal, card or buttons), how they feel, what are the reactions to the robot, what are the critical and positive aspects and so on. For doing both focus groups and observation we have involved two social educational centres:"Fraternità e Amicizia" and "CSE Collage", for a total of 29 participants.

4.1. Hypothesis

Before the testing phase, we made some considerations about the interaction method that we thought would have been the most used and about the results that we would have obtained at the end. Being aware of the fact that often children with NDD, especially ASD, have some problems with speeches and talks, we have hypothesized that buttons and cards would have been the most used interaction paradigms. This because the mapping that can be create between buttons/cards and actions/objects is very simple and easy to understand; there is no need of communication audibly or of the use of correct words and grammar, actions like pressing a button or showing a card speak for themselves.

4.2. Results

At the end of test phase we proceeded with the analysis of the results. We had a mix of raw data that we had to analyze: some of them were sentences that users told during the observations, some others were reports in natural language of the actions of the users, some others were numerical and some others were in survey form. We report here the most important results; in Table 1 there are the results of the second observation. In Figure 5 there are the results of a survey that we made during first focus group. Questions are report here:

1. *Experience with robot*
2. *I would feel relaxed talking to robots*
3. *I would feel uncomfortable if I was given a job where I had to use robots*
4. *Are robots similar to humans?*
5. *Are robots scary?*
6. *Are robots weird?*
7. *Are robots ugly?*
8. *Are robots dangerous?*
9. *Are robots aggressive?*

Example of Table

Person ID	Modality Preferred	Activity Completed	Help Needed
P7	Cards	yes	no
P8	Cards	yes	no
P9	Cards	yes	no
P10	Cards	yes	no
P11	Cards	yes	no
P12	V and C	yes	no
P13	V and C	yes	no

Table 1: Results of the second observation in "CSE Collage" center.

Figure 6 shows a word cloud that represent the main themes which are shared features, ideas, perceptions, pros and cons during the individual activity session.

Figure 6: The word cloud shows the graphical representation of word frequency came out during all observation sessions

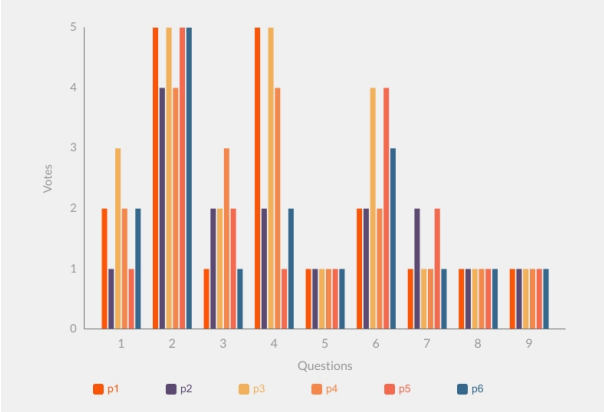


Figure 5: This plot represents the answers of the survey on the first focus group (section 4.2, first set of questions). On the x axis there are the questions, on the y axis there are the possible answers (from 1 to 5) and px are the participant of the focus.

Figure 7: First day of focus group at i3lab of Politecnico with participants from Fraternità e Amicizia

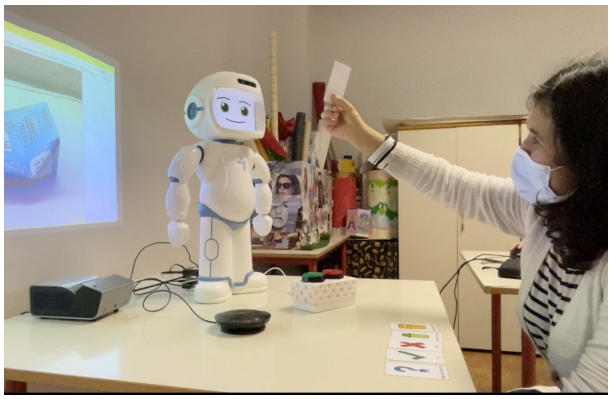


Figure 8: A participant from CSE Collage interacting with the robot using the cards

5. Conclusions

The thesis work shows very important results in the field of interactions between socially assistive robot and people with neurodevelopmental disorders. The main outcome is that traditional communication methods, such as verbal one, don't always fit the need of our target. Some of them are more encouraged to interact with a robot using cards or buttons. Wrapping up results, we can state that *Voice* is a good communication method that works well in a lot of situation but it can't be used in any context and not every person feel comfortable in using it; *Buttons* and *Cards* seems to be promising alternatives for both user experience and information convey. In the end we can state that combining different communication paradigms can be a trump card in the interaction between this type of target and robots, engaging even more the first ones: participants can express their willing in a simpler and clearer way. We hope that our contribution in this field helped to get closer people and technology and made communications easier and within everyone's reach.

References

[1] Imageai repository. `https://github.com/OlafenwaMoses/ImageAI`.

[2] Pytree documentation. `https://py-trees.readthedocs.io/en/devel/index.html`.

[3] Spitale Micol, Birmingham Chris, Swan, R. Michael, and Maja J Matarić. Composing harmoni: An open-source tool for human and robot modular open interaction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[4] Felippe Sartorato, Leon Przybylowski, and Diana K.Sarko. Improving therapeutic outcomes in autism spectrum disorders: Enhancing social communication and sensory processing through the use of interactive robots. *Journal of Psychiatric Research*, pages 1–11, 2017.

Scuola di Ingegneria Industriale e dell'Informazione
Dipartimento di Elettronica, Informazione e Bioingegneria
Master of Science in Computer Science and Engineering

POLITECNICO
MILANO 1863

Multi-modal Communication Interactions Between Socially Assistive Robot and People with Neurodevelopmental Disorders

Advisor: Prof. Franca Garzotto
Co-Advisor: Micol Spitale

Thesis by:
Corrado Pacelli Matr. 10690907
Tharushi Kinkini De Silva Matr. 10537599
Pallimulla Hewa Geeganage

Academic Year 2020–2021

Abstract

Communication plays a vital role in human life in fact it serves five major goals: informing, expressing feelings, imagining, influencing, and meeting social expectations. It could be very difficult for people with Neurodevelopmental Disorders to communicate with others. NDD ranges from 0.4% to 3% of the worldwide population across countries and are characterized by early-onset deficits of variable severity in personal, social, academic, or occupational functioning. Over the years, a number of researches have shown how the support of socially assistive robots can be effective in therapy services. Considering the above mentioned information we have conducted an exploratory research for which we have created a reinforcement social activity with QTrobot where the participants can interact with different modalities: voice, cards and buttons. Our research question wants to investigate on one hand what is the most suitable interaction paradigm for people with NDD while interacting with a robot and on the other to find an efficient way to implement this type of interaction. For answering the first question we have conducted some qualitative research methods: focus groups, observations and surveys. We have involved two social educational centres: Fraternità e Amicizia" and "CSE Collage" for a total of 29 participants. Regarding the implementation we integrated in an existing open-source framework for human robot interaction, HARMONI, behaviour trees using python library, pytrees. This combination allowed us to come up with an efficient activity that comprises qualities of both. The whole research highlights that combining different communication paradigms can be a trump card in the interaction between this type of target and robots, engaging even more the first ones. In fact if we consider the gathered results from the participants, it arose that they would like to do activities with robot and that cards are the preferred interaction paradigm.

Sommario

La comunicazione gioca un ruolo di vitale importanza nella vita quotidiana infatti è usata per: informare, esprimere sentimenti, immaginare, influenzare e soddisfare le aspettative sociali. Può essere tuttavia molto difficile per alcune persone comunicare con gli altri, come nel caso di persone con problemi del neurosviluppo (NDD). Gli NDD sono una fetta non trascurabile della popolazione che spazia tra lo 0.4% e il 3% della popolazione mondiale e sono caratterizzati da deficit di funzioni in ambito sociale, accademico e personale. Con il passare degli anni, un numero sempre maggiore di ricerche ha mostrato che i socially assistive robots possono essere molto utili nell'ambito terapeutico. Considerando le informazioni sopra citate abbiamo condotto una ricerca esplorativa per la quale abbiamo creato un'attività di rinforzo sociale con QTrobot dove i partecipanti possono interagire con diverse modalità: voce, carte e bottoni. Le nostre domande di ricerca sono duplici: da un lato vogliamo investigare sul miglior paradigma di interazione uomo-robot per persone con NDD, e dall'altro vogliamo trovare un modo efficiente per implementarlo. Per rispondere alla prima domanda abbiamo utilizzato alcuni metodi di ricerca qualitativi: focus group, osservazioni, questionari. Abbiamo coinvolto due centri per l'educazione sociale:"Fraternità e Amicizia" e "CSE Collage", per un totale di 29 partecipanti. Per quanto riguarda l'implementazione abbiamo integrato in un framework, già esistente e open-source per l'interazione uomo-macchina, HARMONI, alberi di comportamento usando una libreria di python chiamata pytree. Questa combinazione ci ha permesso di creare un'attività multimodale che comprendesse i vantaggi di entrambi i framework. L'intera ricerca ha evidenziato come la combinazione di diversi paradigmi di interazione possa essere una carta vincente nell'interazione uomo-robot risultando anche più coinvolgente

per il primo. I risultati ottenuti dai partecipanti infatti mostrano che questi ultimi provano piacere nello svolgere attività con i robot e che le carte sono state il paradigma di interazione preferito.

Contents

1 Introduction **1**

1.1 Motivation and Goal 1

1.2 Research questions 2

1.3 Research approach 2

1.4 Research contribution 4

1.5 Thesis structure 5

2 State of Art **7**

2.1 Neurodevelopmental disorders 7

2.1.1 Autism 8

2.2 Human Robot Interaction 12

2.2.1 Socially Assistive Robots 15

3 Design **19**

3.1 User needs 19

3.2 Goal 19

3.3 System requirements 20

3.4 Solution 20

3.5 Human-robot interaction 21

3.5.1 QTrobot 21

3.6 Interaction paradigms 24

3.6.1 Verbal 25

3.6.2 Visual Cards 26

3.6.3 Buttons 28

3.7 Activity designed 29

3.7.1 Main Activity 29

3.7.2 Background 31

4 Implementation **41**
4.1 HARMONI . . . 41
4.1.1 Goals . . . 42
4.1.2 Structure . . . 43
4.2 PyTree . . . 49
4.3 Integration . . . 55
4.4 Body . . . 56
4.5 Leaves . . . 70
4.5.1 ImageAI . . . 70
4.5.2 Amazon Lex . . . 84
4.5.3 Amazon Polly Text-To-Speech . . . 93
4.5.4 Google Speech-To-Text . . . 94
4.5.5 Buttons . . . 96
4.5.6 Scene Manager . . . 99
4.5.7 Subtree Result . . . 103
4.5.8 Web Page . . . 105
4.5.9 Face . . . 105
4.5.10 Speaker and External Speaker . . . 106
4.5.11 Microphone . . . 107
4.5.12 Camera . . . 107

5 Exploratory Research **109**
5.1 Method . . . 109
5.1.1 Study design . . . 110
5.1.2 Questionnaires . . . 119
5.1.3 Participants . . . 120
5.1.4 Tools . . . 126
5.1.5 Hypothesis . . . 128
5.1.6 Data analysis . . . 128
5.2 Results . . . 129
5.3 Discussion . . . 142
5.3.1 Limitations . . . 143

6 Conclusions and Future work **145**
6.1 Conclusion . . . 145
6.2 Future work . . . 146

A PyTree **149**

A.1 Setup 149

A.2 Usage 150

A.3 Testing 151

B Imageai **153**

B.1 Setup 153

B.2 Usage 154

B.3 Testing 154

Chapter 1

Introduction

1.1 Motivation and Goal

"I'm a great believer that any tool that enhances communication has profound effects in terms of how people can learn from each other, and how they can achieve the kind of freedoms that they're interested in."
- Bill Gates

Communication, *the human connection*, is the successful conveying or sharing of ideas and feelings. It plays a vital role in human life in fact it serves five major goals: informing, expressing feelings, imagining, influencing, and meeting social expectations.
During our research we have focused on people with Neurodevelopmental disorders (NDD) that according to the Diagnostic and Statistical Manual of Mental Disorders 5th edition (DSM-5), are characterized by early-onset deficits of variable severity in personal, social, academic, or occupational functioning. The prevalence of intellectual disability disorders typically ranges from 0.4 to 3 percent of the population across countries [1]. They affect approximately 18 million European citizens per year, nearly 4th of the population. Their effects arise during childhood, adolescence, or early adulthood, but are usually life-long [2]. People with NDD, among the various symptoms present also difficulty in communication. The latter often limits them in expressing their ideas, feelings, opinions and in relating to others. It is often taken for granted that human communicates only in a certain way without tak-

ing into account the different characteristics that each one has. Verbal communication is the most used one, however if we think about this target, they often have repetitive or rigid language or uneven language development [3]. So why not facilitate communication, in particular using a robot, easing different interaction modalities thus it is the user himself who decides which is the mode in which he feels most comfortable or he likes best. In order to achieve this we rely on assistive technology which has proved to have great potential in healthcare settings and as mental health interventions for children. Robots have been shown to generate a number of social and behavioral benefits in children with ASD including heightened engagement, increased attention, and decreased social anxiety [4]. Our goal therefore becomes that of investigating the several interaction modalities that a user of the aforementioned target can have with a robot to improve their social skills.

1.2 Research questions

Our research questions wants to investigate on:

- what is the most suitable interaction approach to communicate for people with NDD while interacting with a robot.
- what is an efficient way to enable the implementation of different communication modalities for adapting for different individual needs.

1.3 Research approach

As means to have a better understanding the existing questions, we settled for exploratory research. This latter is often referred to as the building blocks of an overall research project as it used to answer questions like what, why and how. We have emphasised qualitative research methods with the objective to find meanings, observe the target audience behaviour, opinions and preferences. First of all we have created an innovative and positive reinforcement activity that combine Applied behavioural analysis (ABA) therapy with Socially assistive robot

(SAR) giving the possibility to interact with different paradigms: voice, cards and buttons. We came up with a short story on recycling in order to reinforce or teach basic concepts and social skills. After the implementation of the latter, we organized two focus group and three observations with people with NDD.
The focus group (section 5.1.1) comprised each day six, seven participants with NDD but having a fairly high level cognitive skills respect than others with same disorders. During these sessions we had the possibility to collect ideas, opinions, suggestions related our work and most of all on what they would have preferred both on interaction paradigm and activities. Whereas the observations (section 5.1.1) days were based on individual testing of our implemented activity with Qt Robot.
During the observation each participants tried the activity individually and we have collected information on how they prefer to interact with the robot (verbal, card or buttons), how they feel, what are the reactions to the robot, what are the critical and positive aspects and so on.
Regarding the implementation we needed a framework that gave us the possibility to:

1. easily add components for the different interaction methods

2. face the challenge of real time computing [1]

The framework used is HARMONI, that is a comprehensive tool containing components and capabilities that developers can use in order to get a social assertive up and running on a robot. Since we also needed a high degree of flexibility in the choice of the execution path to be executed at run time, we decided to enchant HARMONI by empowering it with behavioural trees. A behavior tree is a mathematical model of plan execution used in computer science, robotics, control systems and video games. They describe switching between a finite set of tasks in a modular fashion [6].

[1]Real-time computing is the computer science term for hardware and software systems subject to a ”real-time constraint”. Real-time programs must guarantee response within specified time constraints [5].

To integrate behaviour tree in HARMONI we used PyTree, a python library that implements all the components present in the theory of behaviour tree.

1.4 Research contribution

We can identify two research contribution:

- **design and method**: we have designed an immersive activity that combines both Applied behavioural analysis (ABA) therapy with Socially assistive robot (SAR) giving the possibility to interact with different paradigms that are verbal, cards and buttons. In particular, given a specific target, we tried through an exploratory research to identify what is the most suitable communication approach, why it is preferred, what is the background of the people concerned and so on. As a result of the focus groups and observations, it emerged that cards was the most preferred interaction paradigm, confirming our hypothesis.
  Giving the participant different interaction modalities, these allow them to choose the preferred one and what make them most at ease to communicate. The use of other communication tools such as cards and buttons have improved the engagement and the interaction.

- **implementation**: we tried to figure out how to facilitate these interaction paradigms from development point of view.
  We have implemented inside an open source, Robot-Agnostic, modular and composable framework that is HARMONI, behaviours trees. We have integrated python library for trees and it gives several advantages like simplicity, scalability,modularity, dynamism and priority handling.
  Moreover we have add a new module in HARMONI that is able to perform image recognition using ImageAI python library.

1.5 Thesis structure

- **Chapter 1 - Introduction**: provides a brief overview of the main topics covered in the thesis. It gives a big picture of the motivations, the purpose, the research method and the contributions provided.
- **Chapter 2 - State of Art**: comprises the whole background of Neurodevelopmental disorders (NDD), autism and existing therapies and the related literature of Human Robot Interaction (HRI) and Socially Assistive Robot (SAR).
- **Chapter 3 - Design**: review the whole design phase of the proposed activity going into details of user need, goal, system requirements, solution, HRI and activity designed.
- **Chapter 4 - Implementation**: goes into depth on the concretization of what we had conceived, in particular it shows what we have used to better realise it. We started from HARMONI and then we end up integrating in it python behaviour tree.
- **Chapter 5 - Exploratory Research**: describe what typologies of research methods we have chosen and conducted. This chapter is the most important to answer to our first research question regarding the most suitable communication approach for our specific target.
- **Chapter 6 - Conclusion and Future Work**: it concludes the thesis wrapping up all the considerations about the entire work we have done, focusing on the results, what we have achieved so far, what are the pros and cons and what could be the future work and improvements.
- **Appendix**: it contains one documentation that explains how to integrate pytree in HARMONI and another one on how to use ImageAI modules.

Chapter 2

State of Art

2.1 Neurodevelopmental disorders

Neurodevelopmental disorders, according to the Diagnostic and Statistical Manual of Mental Disorders 5th edition (DSM-5), are characterized by early-onset deficits of variable severity in personal, social, academic, or occupational functioning [7]. Examples of Neurodevelopmental disorders in children include attention-deficit/hyperactivity disorder (ADHD), autism, learning disabilities, intellectual disability (also known as mental retardation), conduct disorders, cerebral palsy, and impairments in vision and hearing. Children with neurodevelopmental disorders can experience difficulties with language and speech, motor skills, behavior, memory, learning, or other neurological functions. While the symptoms and behaviors of NDD disabilities often change or evolve as a child grows older, some disabilities are permanent. Diagnosis and treatment of these disorders can be difficult; treatment often involves a combination of professional therapy, pharmaceuticals, and home- and school-based programs [8]. Most neurodevelopmental disorders have complex and multiple contributors rather than any one clear cause. These disorders likely result from a combination of genetic, biological, psychosocial and environmental risk factors [9, 10, 11, 12, 13, 14, 15, 16].

2.1.1 Autism

Symptoms

Autism spectrum disorder (ASD), or autism, is a broad term used to describe a group of neurodevelopmental conditions. Children with ASD start to show symptoms at an early age. The symptoms continue during childhood and adulthood. The last mentioned typically become clearly evident during early childhood, between ages 12 and 24 months. However, they may also appear earlier or later. Early symptoms may include a marked delay in language or social development. The DSM-5 includes the following symptoms of ASD:

- problems with communication and social interaction
- restricted or repetitive patterns of behavior or activities

To be diagnosed with autism, a person must experience symptoms in both of these categories [17].
Social communication and interaction skills can be challenging for people with ASD. Examples of social communication and social interaction characteristics related to ASD can include [17]:

- Avoids or does not keep eye contact.
- Does not respond to name by 9 months of age.
- Does not show facial expressions like happy, sad, angry, and surprised by 9 months of age.
- Does not play simple interactive games like pat-a-cake by 12 months of age.
- Uses few or no gestures by 12 months of age (e.g., does not wave goodbye).
- Does not share interests with others (e.g., shows you an object that he or she likes by 15 months of age).
- Does not point or look at what you point to by 18 months of age.
- Does not notice when others are hurt or sad by 24 months of age.
- Does not pretend in play (e.g., does not pretend to “feed” a doll by 30 months of age) Shows little interest in peers.

- Has trouble understanding other people's feelings or talking about own feelings at 36 months of age or older.
- Does not play games with turn taking by 60 months of age.

People with ASD have behaviors or interests that can seem unusual. These behaviors or interests set ASD apart from conditions defined by only problems with social communication and interaction.
Examples of restricted or repetitive interests and behaviors related to ASD can include [17]:

- Lines up toys or other objects and gets upset when order is changed.
- Repeats words or phrases over and over (i.e., echolalia).
- Plays with toys the same way every time.
- Is focused on parts of objects (e.g., wheels).
- Gets upset by minor changes.
- Has obsessive interests.
- Must follow certain routines.
- Flaps hands, rocks body, or spins self in circles.
- Has unusual reactions to the way things sound, smell, taste, look, or feel.

Most people with ASD have other characteristics. These might include [17]:

- Delayed language skills.
- Delayed movement skills.
- Delayed cognitive or learning skills.
- Hyperactive, impulsive, and/or inattentive behavior.
- Epilepsy or seizure disorder.
- Unusual eating and sleeping habits.
- Gastrointestinal issues (e.g., constipation).
- Unusual mood or emotional reactions.
- Anxiety, stress, or excessive worry.
- Lack of fear or more fear than expected.

Therapies

There's no cure for autism, but several therapeutic approaches can help to improve social functioning, learning, and quality of life for both children and adults with autism. Since autism is a spectrum-based condition, some people may need little to no support, while others may require intensive therapy [18].
It's also important to keep in mind that a lot of the research related to support for autism focuses on children. This is largely because existing research suggests that support is most effective when started before age 3. Research has shown that early intervention can improve a child's overall development. Children who receive autism-appropriate education and support at key developmental stages are more likely to gain essential social skills and react better in society. Essentially, early detection can provide an autistic child with the potential for a better life [19]. Still, many of the options designed for children can help adults as well.
These are some therapies that can help them feel better or alleviate certain symptoms [18]:

- **Applied Behaviour Analysis (ABA)**: formally known as the Lovaas Model, is one of the most widely used interventions for the treatment of ASD and has been proven to be effective [20]. ABA "has the distinction of having the longest history and the most extensively documented evidence base to support its efficacy in the treatment of autism" [21]. ABA refers to a set of principles that focus on how behaviors change, or are affected by the environment, as well as how learning takes place. The term behavior refers to skills and actions needed to talk, play, and live. While these principles impact everyone each day, they can be applied systematically through interventions to help individuals learn and apply new skills in their daily lives. The ultimate goal of ABA is to establish and enhance socially important behaviors. Such behaviors can include academic, social, communication, and daily living skills; essentially, any skill that will enhance the independence and/or quality of life for the individual [22].

- **Cognitive behavioral therapy (CBT)**: is a type of talk therapy that can be effective in helping children and adults. During CBT sessions, people learn about the connections between feelings, thoughts, and behaviors. This may help to identify the thoughts and feelings that trigger negative behaviors [18].

- **Social skills training (SST)**: is a way for people, especially children, to develop social skills. For some people with autism, interacting with others is very difficult. This can lead to many challenges over time. Someone undergoing SST learns basic social skills, including how to carry on a conversation, understand humor, and read emotional cues [18].

- **Sensory integration therapy (SIT)**: people with autism are sometimes unusually affected by sensory input, such as sight, sound, or smell. Social integration therapy is based on the theory that having some of your senses amplified makes it hard to learn and display positive behaviors. SIT tries to even out a person's response to sensory stimulation. It's usually done by an occupational therapist and relies on play, such as drawing in sand or jumping rope [18].

- **Occupational therapy (OT)**: is a field of healthcare that focuses on teaching children and adults the fundamental skills they need in everyday life. For children, this often includes teaching fine motor skills, handwriting skills, and self-care skills. For adults, OT focuses on developing independent living skills, such as cooking, cleaning, and handling money [18].

- **Speech therapy**: teaches verbal skills that can help people with autism communicate better. It's usually done with either a speech-language pathologist or occupational therapist. It can help children improve the rate and rhythm of their speech, in addition to using words correctly. It can also help adults improve how they communicate about thoughts and feelings [18].

2.2 Human Robot Interaction

Human–Robot Interaction (HRI) is a field of study dedicated to understanding, designing, and evaluating robotic systems for use by or with humans. Interaction, by definition, requires communication between robots and humans and it may take several forms [23].
As it happens, if a non-researcher interacts with a robot that he or she has never encountered before, then what matters is how the robot looks, what it does, and how it interacts and communicates with the person. The 'user' in such a context will not care much about the cognitive architecture that has been implemented, or the programming language that has been used, or the details of the mechanical design. Studying interactions with robots and gaining general insights into HRI applicable across different platforms is therefore a big challenge [24].
Scholtz provided a taxonomy of roles that robots can assume in HRI [25]:

- Supervisor,
- Operator,
- Mechanic,
- Peer or Companion and
- Bystander

To this list, we add the following:

- Coach: the robot is in a teaching or leadership role for the human
- Information Consumer: the human does not control the robot, but the human uses information coming from the robot in, for example, a reconnaissance task.

Table 2.1 classifies the most frequent types of interactions for some application areas.
As concerns socially relevant research, robots are being developed to serve in assistive and educational capacities. Assistive robotics is perhaps one of the highest profile areas of HRI in the world. This application domain often places the robot in a peer-like or mentoring role

with the human in practice, even though the intention of the robot is designed to provide service to the human [24].
For some people with physical and mental challenges, robots may provide an opportunity for interaction and therapy. Such work is being explored with autistic children [26, 27]. Many of these children respond weakly or not at all to social cues, but respond well to mechanical devices. Robots provide a possible therapeutic role for using a mechanical device to improve social interactions [28].

Application Area	Role	Example
Search and rescue	Human is supervisor or operator	Remotely operated search robots
	Human and robot are peers	Robot supports unstable structures
Assistive robotics	Human and robot are peers, or robot is tool	Assistance for the blind, and therapy for the elderly
	Robot is a coach	Social interaction for autistic children
Military and police	Human is supervisor	Reconnaissance, de-mining
	Human and robot are peers	Patrol support
	Human is information consumer	Commander using reconnaissance information
Edutainment	Robot is a coach	Robotic classroom assistance
	Robot is a coach	Robotic museum tour guide.
	Robot is peer	Social companion
Space	Human is supervisor or operator	Remote science and exploration
	Human and robot are peers	Robotic astronaut assistant
Home and industry	Human and robot are peers	Robotic companion
	Human is supervisor	Robotic vacuum
	Human is supervisor	Robotic construction

Table 2.1: Examples of roles that arise in several application areas

2.2.1 Socially Assistive Robots

Socially Assistive Robots (SAR) is a relatively new field in robotics, aiming at assisting users through social rather than physical interaction. SAR focus to elicit the proper social, emotional and cognitive cues to encourage the involved target groups, including patients in rehabilitative therapies, elderly population and individuals with developmental disorders, such as autism [29, 30, 31, 32]. Regarding the latest category, SARs offer the potential to support them in educational, social and behavioral interventions, obtaining different roles during therapies.

A number of studies suggest that children with autism can be more comfortable interacting with robots, possibly because the robots' behavior and reactions are more consistent and predictable than that of humans [33, 34, 35, 36, 37]. With the increase in rates of individuals diagnosed with autism spectrum disorder, there is an increasing need for reasonable and accessible applied behavior analysis (ABA) therapy services. Early diagnosis and regular therapeutic intervention are critical factors impacting the potential impact of therapy with children on the spectrum. However these services are not universally accessible nor affordable, there are some barriers for sufficient access to therapy include high administrative burden, burnout rates on the part of ABA therapists, and affordability of services. One way to reduce administrative burden is to automate procedures using computer-based interventions, including SARs. SARs have the potential increase user engagement, while at the same time making it possible for therapists to provide a more interactive session for their user [30, 38].

Specifically, long-term SAR interventions provide the benefits of personalization [39],reduce the effects of novelty and discomfort [40] and strengthen human-robot attachment [41]. The referred considerations indicate that the design of Socially Assistive Robots must be attentive, due to their implementation in a very specific and vulnerable population. Robotics researchers should develop SAR paying attention to specific needs that are Engagement, User Perception (perceived sociability, social presence, trust,anxiety, likability) and requirements of these children and bearing in mind recommendations by therapists [42]:

- **Appearance**: it is strongly believed that robot's appearance is an impact factor which influence both the behavior of the child and the outcomes of the intervention [43, 44, 33]. Autism studies have been conducted with different types of robots such as robots with human-like appearance, animal-like robots, mascot-type robots and mechanical-type robots where, each type of robot has its strengths and drawbacks in interventions with ASD children. A level of congruency between robots' appearance and capabilities should exist, otherwise children will create false expectations and the interaction will not have the desired [43]. It can be said that robots are possible to be accepted, if their appearance and their overall design are plain and simple.

- **Empathy**: Another key aspect of accepting robots in autism interventions is their empathetic behavior. Due to children's nature, which is characterized by isolation and disturbance, SAR need to be as much close as they can to children's states to assist them successfully. Empathy is the ability to feel or to feel into somebody or something and robots lack this ability. Empathy is mentioned as important factor, because robots should display interest towards children during interaction. This could be achieved by affective feedback and responses accordingly to child's behavior [45]. Robots should be equipped with skills, capable enough to communicate their own states, to encourage children's emotional expression and motivate them to complete a task.

- **Autonomy**: Autonomy is another consideration that affects SAR's behavior and thus should be considered. Some researchers claim that robots should have a sufficient level of autonomy, otherwise therapists will concentrate entirely in controlling the robot, rather than assisting the child [46, 47]. Towards this direction, a semi-autonomous robot will allow therapists to adjust its behavior in accordance to children's states [43, 30].

- **Physical**: Finally, the use of physical robots in interventions instead of digital agents, indicating the benefits of involving em-

bodied robots in autism interventions. Physical robot interactions are more engaging and credible in collaborative tasks, offering encouragement and guidance to children with autism [48, 49].

In terms of use in ABA settings, however, there is a need for a conceptually systematic understanding of how and why certain aspects of the SAR either do or do not work effectively to carry out, or assist in, the delivery of therapy. Since ABA therapists have a limited number of hours of availability in a given time period (e.g., per day or per week),developers of ABA-related SARs hope to fill time gaps in treatment by creating robots that can be used either autonomously or remotely to assist therapy.

That is not to say a robot will replace humans in the intervention, but if the social robot leads to the child spending more time on the task at hand, then there should be greater gain in skills, rather than time spent redirecting a child who may be described as unruly, inattentive, or tantruming [30].

Overall, these technology interventions, such as SARs, may be an affordable and effective option that should be incorporated to increase access and affordability and consistency of ABA services, and to reduce the administrative burden on therapists. Moreover if they are also portable, and increase engagement, then they could be used in the schools and at home, providing an intervention tool that may increase therapy hours without increasing the cost.

Chapter 3

Design

3.1 User needs

Children with NDD need to develop social skills, self-care skills and academic abilities.
It is important to help children to express themselves with ease considering that sometimes they may need different type of communication than verbal one. They need also to understand how others might behave or respond in a particular situation and increase interaction capability with others. The users need activity and environment more immersive and welcoming. They may only be able to handle one thought or idea at a time so conversations has to be focused and simple.
Children with NDD may have trouble showing their feelings. Children may sometimes interpret feedback as an attempt to control and judge them, they can feel drained, stressed, withered, diminished, or worthless. Essential is to provide positive reinforcement about an area of strength or achievement in order to develop self-esteem, being specific about what behaviour they're being praised for.

3.2 Goal

Many children can learn to communicate and interact, what matters is to find the right key in order to make it possible. Our goal is to investigate, among the several interaction modalities, which are the most suitable for this target and from a technological point of view of to real-

ize this communication. We would like to improve the critical aspects of children with NDD, in particular autism, through an innovative and positive reinforcement activity that combine Applied behavioural analysis (ABA) therapy with Socially Assistive Robot (SAR).

3.3 System requirements

System requirements are strictly based on the user needs that are considered to be the first and the most informal ones. We have found as system requirements the followings:

- Using new interaction paradigm based on behaviour and verbal interaction of the child.
- Give the possibility to therapist/host to select different interaction paradigm to better meet children needs.
- Give the possibility to interact verbally
- Give the possibility to interact with the robot using ”magic cards”
- Give the possibility to interact with the robot using buttons
- Adding a greeting phase at the beginning so the participant gets familiar with the agent from the first moment.
- Adding a phase in which the robot explain the context of the story and all the steps of the activity
- Using projector to make the activity more immersive.
- Create a simple instructive and entertaining story
- Provide feedback through the robot

3.4 Solution

Our activity takes inspiration on social stories and it foresees an interaction between robot and child. In particular we built a little story on recycling. Even if the story is fixed, it can slightly change based on

the answers/ choices of the child, so we designed some different path. The focus of the activity is on the way in which the child interact and express his/her knowledge in daily routine activities. Moreover we want to educate and raise awareness of the child so if he/she answers 'in a wrong way' at one or more questions we help him/her in understand the correct answer.
The main characteristics of the designed solution is the possibility to have three different interaction paradigms : verbal, cards and buttons. In this way the child can use the interaction that make him/her feel more comfortable or the therapist can choose the one that deems most appropriate for the child.

3.5 Human-robot interaction

The interaction between robot and child is thought to be very friendly, we would like that the child feels calm and at ease as he is interacting with a friend. The robot is not a judge, it is a companion inside the story that is interested on child's improvement and behaviour.
The robot aside from dialogue, pays close attention, thanks to its incorporated camera, that the child falls within its field of vision.
We have also predicted that the child can stop interacting for some reason, in that case after two times that the robot doesn't receive any kind of feedback from the child, asks him if there is something wrong.

3.5.1 QTrobot

We have used QTrobot for our activity implementation. QTrobot [50] is an expressive social robot designed to increase the efficiency of special need education by encouraging an active and engaged interaction and participation of children with autism (Figure 3.1).
Benefits of using QTrobot for children with autism:

- Attractive and motivating: QTrobot is an engaging technological tool which can be attractive for children with autism. Using QTrobot results in more attention and concentration of children, helping them to learn more effectively, especially for the activities that require shared attention.

- Simple, exaggerated and easy to understand: QTrobot behaves in a very simple and understandable manner. Exaggerated and clear expressions and behaviours of QTrobot help children learn easily. With QTrobot, children can practice their skills in a simplified setup and gain confidence to apply them in their daily life.

- Constant and predictable: By showing a consistent, positive and patient attitude, QTrobot makes children feel more relaxed during interactions and education sessions. The predictable nature of QTrobot gives comfort to children and prevents them from getting distracted or overwhelmed.

- Tireless and non-judgmental: Interaction with QTrobot is free of social demands. It is non-judgmental and never gets tired or frustrated. QTrobot repeats everything in the same manner until a child masters a skill. QTrobot offers a coherent, consistent and positive setup for children to practice new skills as many times as needed.

Figure 3.1: QTrobot: Humanoid Social Robot [50]

Scientific research shows (Figure 3.2) that when QTrobot is used in educational sessions, children have better engagement and less disruptive and stereotypical behaviors, compared to the sessions conducted by an educator alone. The increased attention and reduced anxiety can improve learning outcomes for children with Autism Spectrum Disorder.

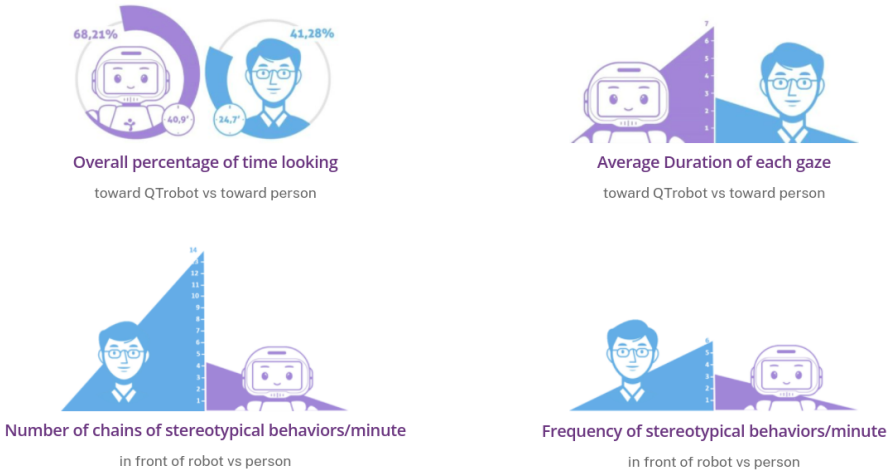


Figure 3.2: QTrobot Scientific results [50].

QTrobot as an embodied technology stands somewhere between a computer and a social being. It offers various benefits of technology such as consistency, simplicity and predictability, but it also has a character providing an engaging environment for social learning. By being a point of joint attention between the child and educator, QTrobot can facilitate a triangular interaction to elicit positive behaviours and promote learning in children with ASD [50] (Figure 3.3).

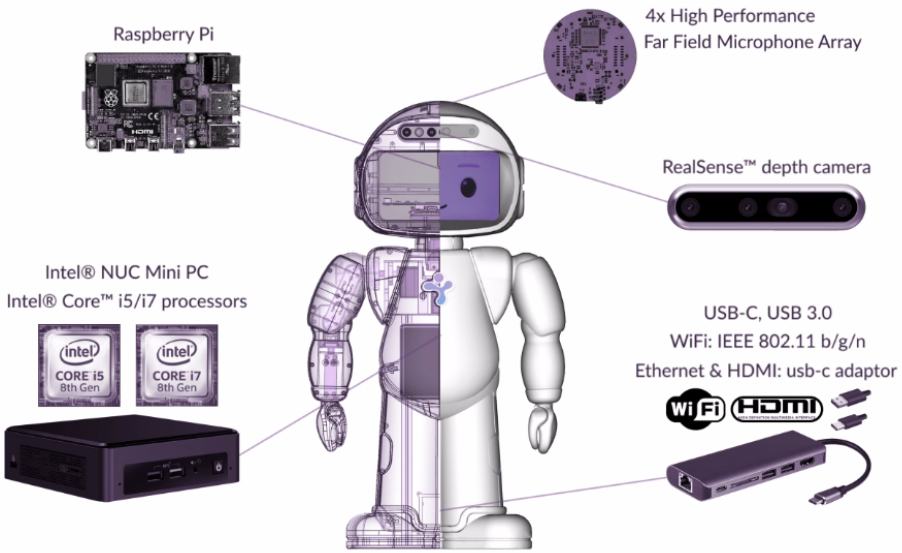


Figure 3.3: QTrobot hardware components [50]

3.6 Interaction paradigms

We have designed three interaction paradigms in order to facilitate communication between the user and the robot whereas not everyone communicates in the same way. The ability of people with NDD to communicate and use language depends on their intellectual and social development. Some children may not be able to communicate using speech or language, and some may have very limited speaking skills. Others may have rich vocabularies and be able to talk about specific subjects in great detail. Other may want to communicate through prompt cards or buttons rather than voice.
In order to accomplish everyone needs and capabilities we tried to create a very simple story where the conversation is basic, characterized by question from the robot that are easy to understand and always linked to a previously explained concept. We have foreseen short answers, since the questions expect one correct answer among two or maximum three possibilities. We tried to design the question and answer session in the most appropriate way using Amazon Lex (section 4.5.2). We tried to figure out what responses the user might give and

how to react without criticizing or judging him but rather conducting him to the correct answer or encouraging him.

3.6.1 Verbal

Verbal communication is the most used one among people. With verbal communication the only way to send a message is vocally, and the only way to receive it is through hearing it audibly.
There are some patterns of language use and behaviors that are often found in children with ASD [3]:

- **Repetitive or rigid language**: Often, children with ASD who can speak will say things that have no meaning or that do not relate to the conversations they are having with others. Or a child may continuously repeat words he or she has heard—a condition called echolalia.Some children with ASD speak in a high-pitched or sing-song voice or use robot-like speech. Other children may use stock phrases to start a conversation [3].

- **Narrow interests and exceptional abilities**: Some children may be able to deliver an in-depth monologue about a topic that holds their interest, even though they may not be able to carry on a two-way conversation about the same topic [3].

- **Uneven language development**: Many children with ASD develop some speech and language skills, but not to a normal level of ability, and their progress is usually uneven. For example, they may develop a strong vocabulary in a particular area of interest very quickly. Many children have good memories for information just heard or seen. Some may be able to read words before age five, but may not comprehend what they have read. They often do not respond to the speech of others and may not respond to their own names. As a result, these children are sometimes mistakenly thought to have a hearing problem [3].

- **Poor nonverbal conversation skills**: Children with ASD are often unable to use gestures—such as pointing to an object—to give meaning to their speech. They often avoid eye contact,

which can make them seem rude, uninterested, or inattentive. Without meaningful gestures or other nonverbal skills to enhance their oral language skills, many children with ASD become frustrated in their attempts to make their feelings, thoughts, and needs known. They may act out their frustrations through vocal outbursts or other inappropriate behaviors [3].

3.6.2 Visual Cards

Many young children with autism spectrum disorder (ASD) do not use words or even imitate sounds the way typically developing children do. Therefore communication enhancement particularly through picture cards should be a major focus of education and intervention [51].
Visual communication has a lot of benefits:

- Visuals can help teach social skills that can be transferred and used within their own social situations.
- The level of attention increases in children with autism while communicating using cue cards/picture cards. Pictures provide an opportunity to view minute details and since children with autism are visual learners it enables them to remember the information perceived. [51]
- Visual aids through pictures improve memory and help high functioning autism to recognize and recall. [51]
- Visual learning through picture cards improves eye contact in children with Autism. [51]
- Autism communication cards help with difficulties such as how to start a conversation and how to respond to others when they make certain social approaches. This could include harnessing the ability to listen more carefully to other people's topics, following instructions, or taking part in activities.
- Children with Autism may not be able to express how they are feeling, or what they want. These visuals can help to prevent problematic behaviours spun from any communication barriers

they have with you. The cards focus on positive routes of communication and relationship building.

Hence to enable children with autism to develop communication, a visual mode (picture cards/cue cards) was prepared. In particular we have prepared eight visual cards with a picture in the center and below, its written explanation. These cards (Figure 4.26) represent the possible answers that the user can give to the robot during the activity session:

- **bidone plastica**: plastic dustbin where all the plastic trash should be thrown.
- **bidone vetro**: glass dustbin where all the glass trash should be thrown.
- **bidone carta**: paper dustbin where all the paper trash should be thrown.
- **sì**: yes card, used to answer for confirmation questions.
- **no**: no card, used to answer for confirmation questions.
- **non ho capito**: i don't understand, used to repeat the previous asked question or said concept.
- **stop**: used to stop the activity session and call the therapist in order to solve the child problem.
- **ombrello**: umbrella, it was created to indicate the need of an umbrella in a particular situation but then we decided to do not show it to the user and change the question from "*It's raining, what do you think we should bring?*" to "*It's raining, do you think we should bring with us an umbrella?*"

Figure 3.4: This figure shows the cards of the activity.

3.6.3 Buttons

We have chosen to use also buttons considering that maybe for some children is more engaging and simple to press a button then speak or showing a card for giving an answer. This because sometimes some mental disorders can come with speech disorder or reluctance in speaking. Some other disorders can attack the motor system hence the fewer movements users have to do, the easier will be the activity; the movements required to press a button are less that the movements required to show a card. Of course it depends on the activity and how many possibilities there are for every question/dialogue in fact you have to associate clearly each response to a button. Maybe it could be better to have different colors or associated sounds.
For our activity we have used just two buttons, red and green. This because we have thought about it at a later stage trying to embrace also children that don't want or can't use neither speech nor cards to communicate. The green button is used to say yes while the red one is used to say no (Figure 3.5).

Figure 3.5: Buttons used for our activity

3.7 Activity designed

We have built a short story on recycling that focuses on paper,plastic and glass. In order to have a more realistic session we have created a story that is similar to a typical morning routine in which they are going to have breakfast and then clear the table and throw the garbage in the correct bin. Everything is accompanied by the projection on the screen or wall to make an immersive story.
The whole session can be divided into two correlated parts: main activity, background.

3.7.1 Main Activity

It is the part related to the story with its teachings. The story has an introduction part where the robot teaches what is the recycling with some examples and then asks the child where to throw some rubbish. Then there is an interlude where they have to go out to buy milk; this part is created to see the responsiveness and how the child behaves in some circumstances. In our activity we thought to show a cloudy and rainy sky and ask the child if he thinks that they should bring an umbrella with them while leaving home.
The story ends with the child and robot going home to have finally

breakfast. This latter takes up the issue of the recycling and asks again the child, to reinforce the concepts, where to throw the garbage. The activity start and end with the same arguments for the purpose of enhance the knowledge acquired.
For better understanding from Figure 3.6 to Figure 3.11 there is represented the diagram of the main activity.
We assume that the answers given by the child include all the different types of interaction: spoken, buttons or cards.
During the design phase we have also identified the modules of the behaviour tree (Figure 3.12).

3.7.2 Background

The background is the component designed to handle some unexpected action of the child during the main activity. Monitoring measures are done in a precise order (Figure 3.13): we first perform the visual checking for which we have given the highest priority and then we continue with the interaction one if the previous one was successfully.

- **visual checking**: is responsible for seeing if the child falls within the robot's field of view. If the child is not there the robot will ask the child where is he and to come in front of it.
- **interaction checking**: checks if the child has no interaction for more than two times and then asks him if there's something wrong.

Every time a background checking is activated because either the child is not there or he is not interacting, the main activity is paused and the above mentioned actions are done. If the problem is solved, the main activity is resumed otherwise we call the therapist and stop the whole session. In Figure 3.14 and Figure 3.15 (page 40) there are the modules that we used in both backgrounds.

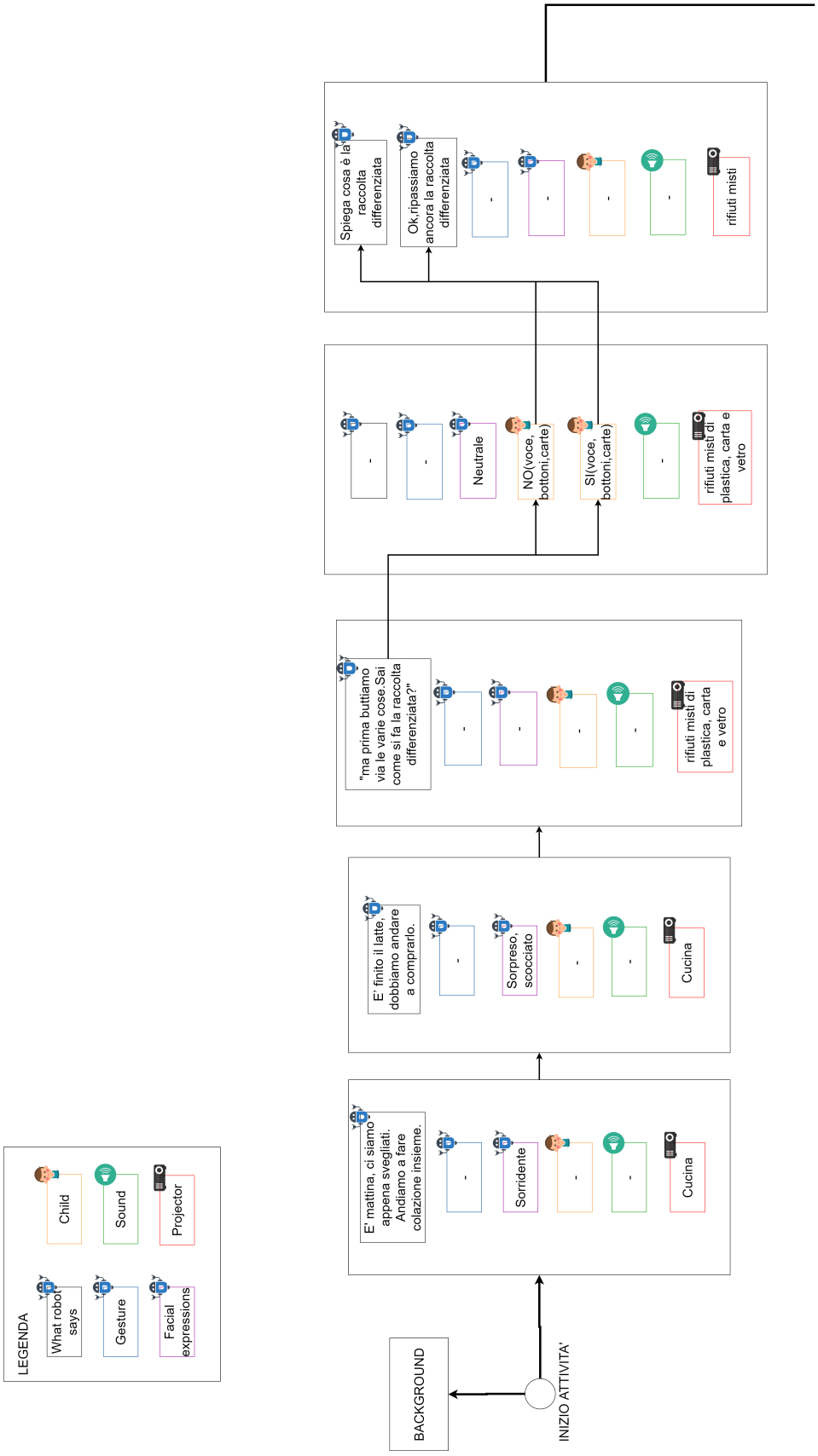


Figure 3.6: First part of the main activity diagram.

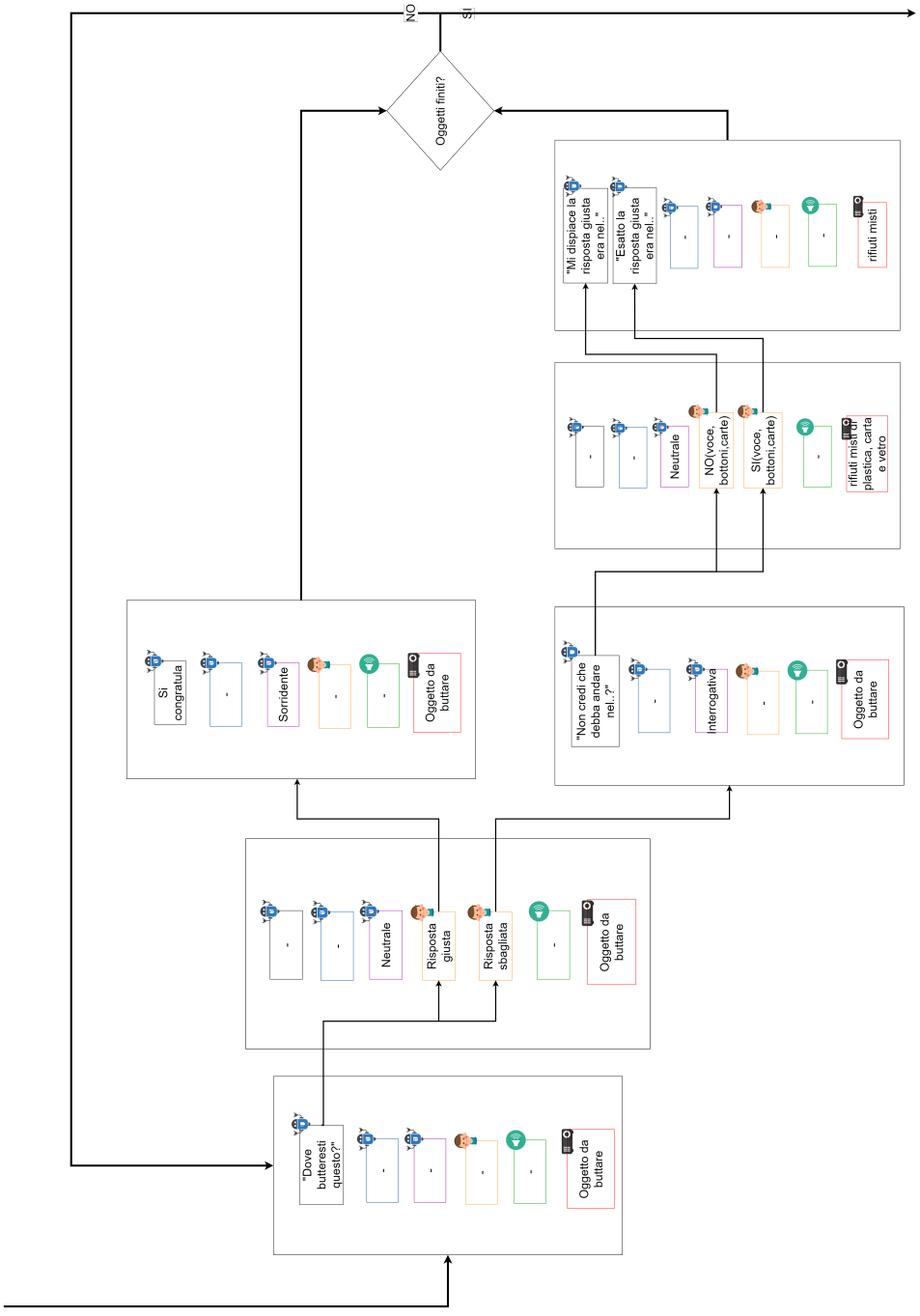


Figure 3.7: Second part of the main activity diagram.

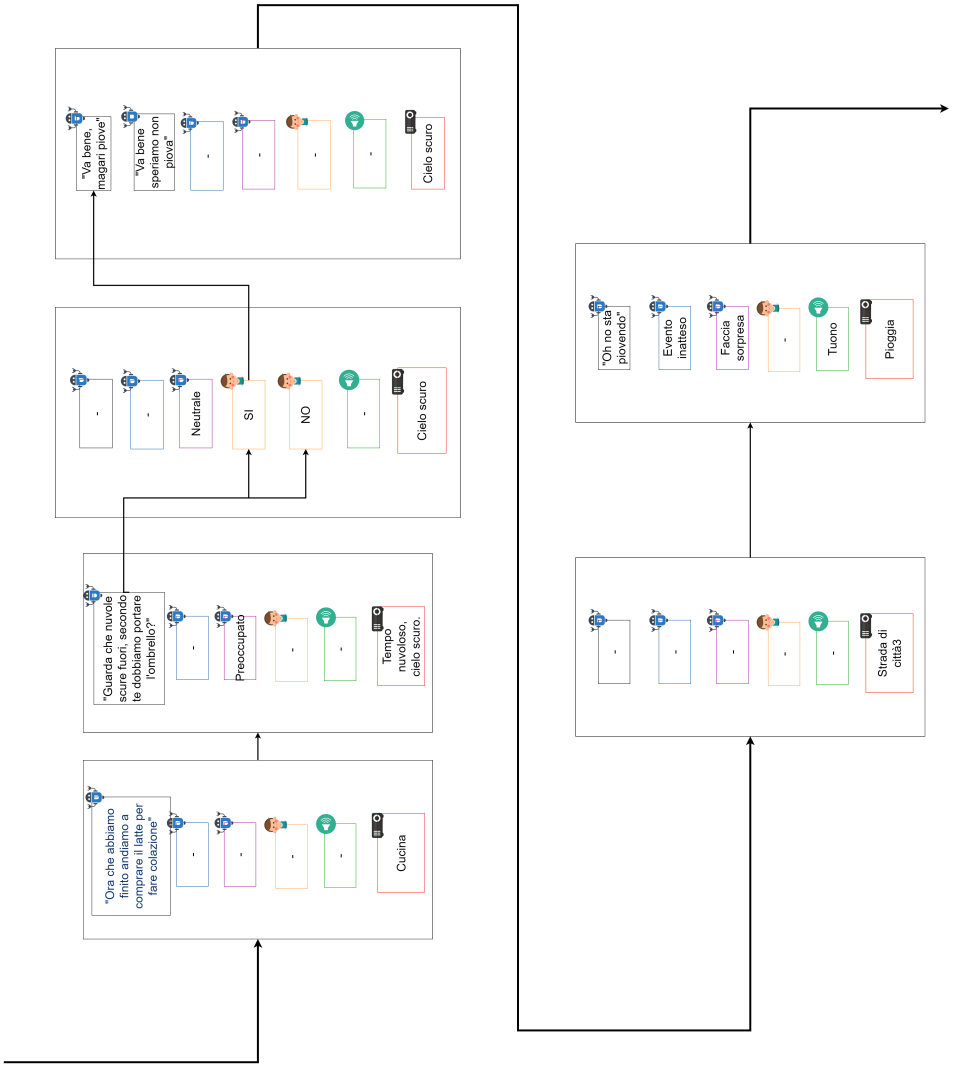


Figure 3.8: Third part of the main activity diagram.

Figure 3.9: Fourth part of the main activity diagram.

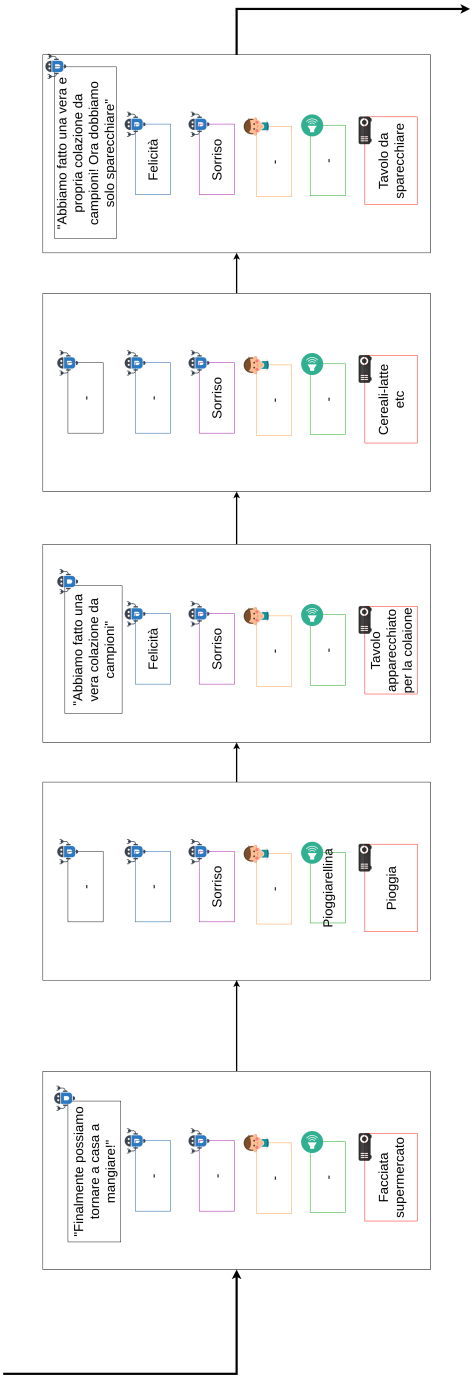


Figure 3.10: Fifth part of the main activity diagram.

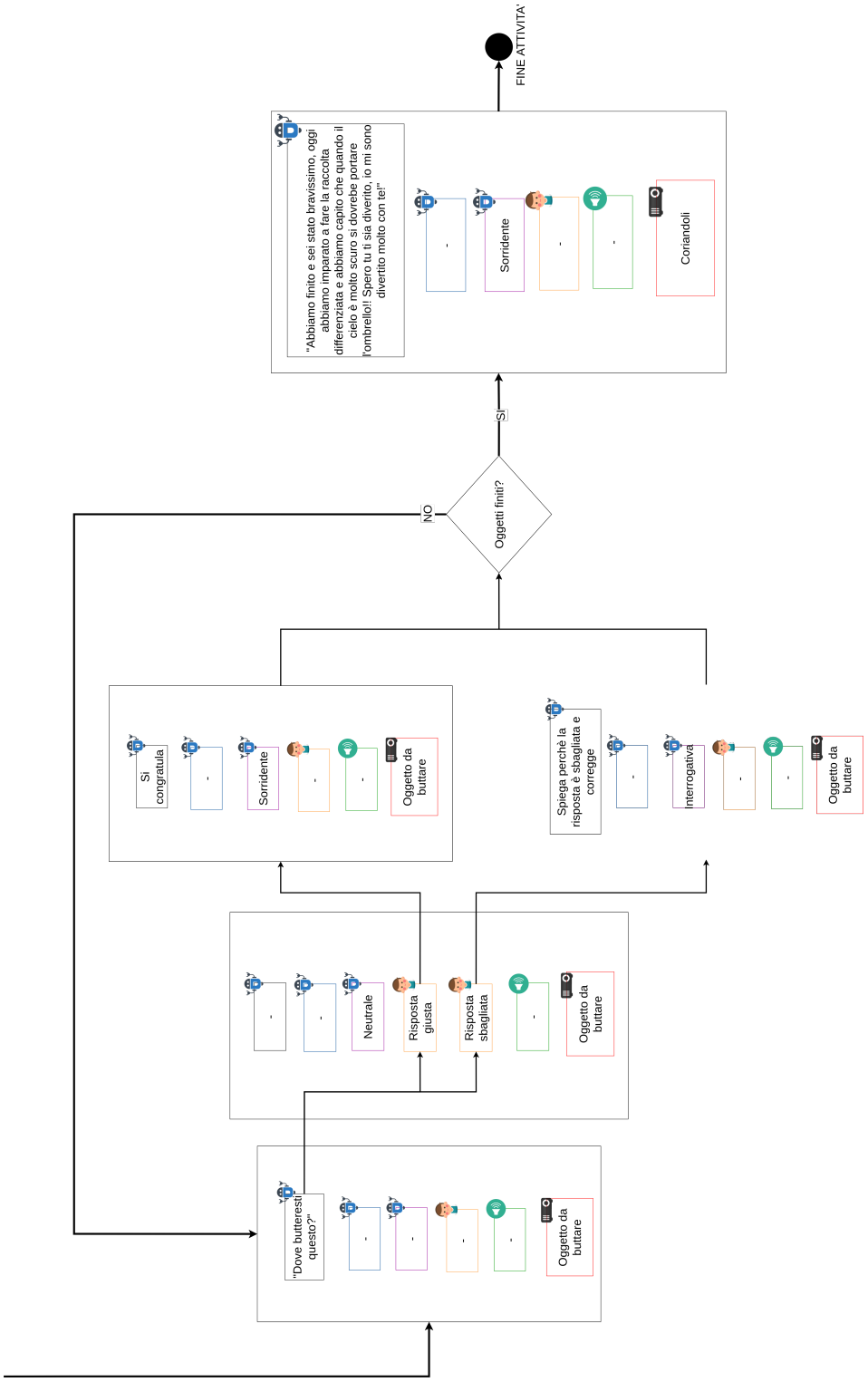


Figure 3.11: Sixth part of the main activity diagram.

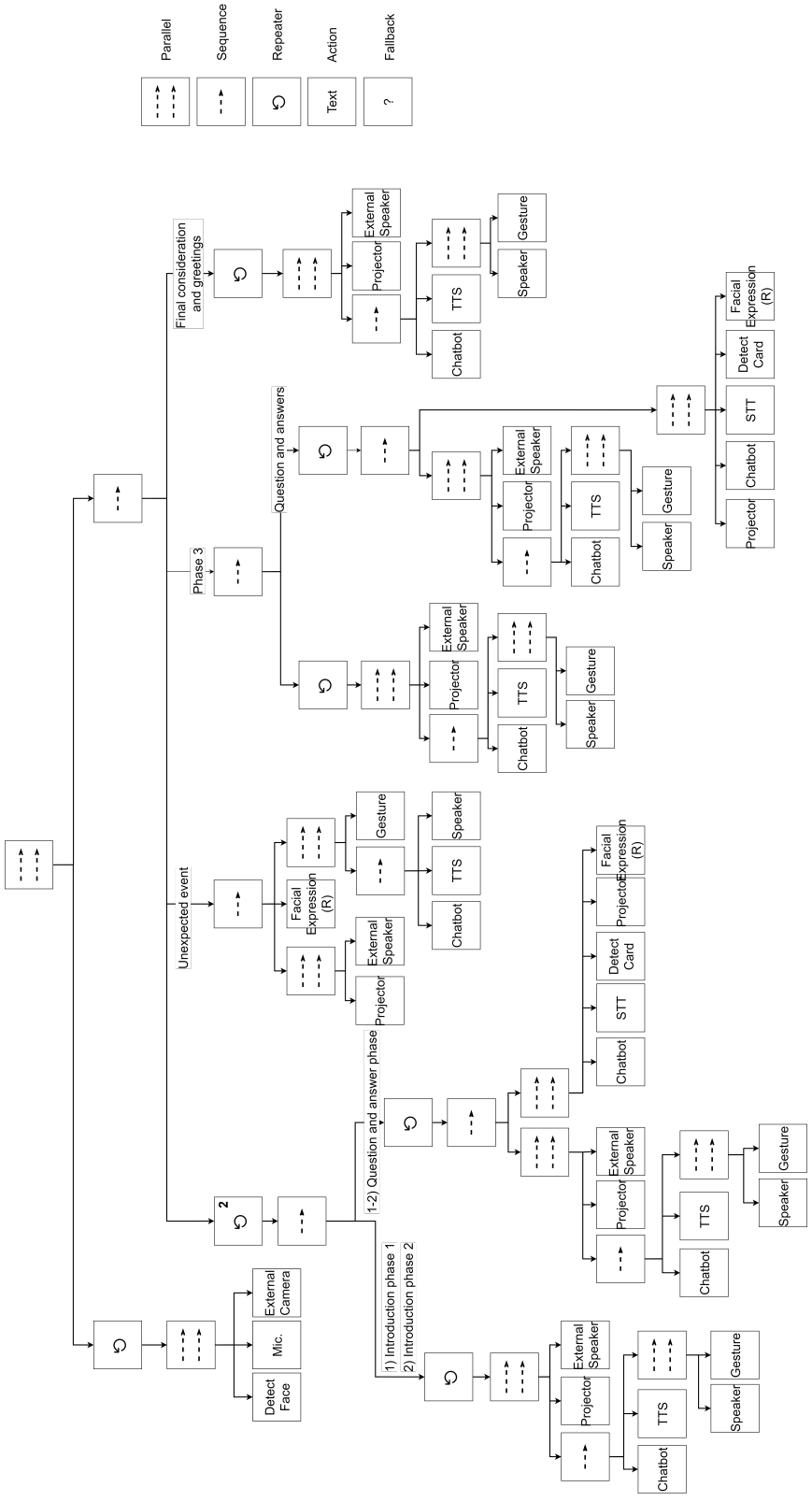


Figure 3.12: Diagram representing the modules of the main activity

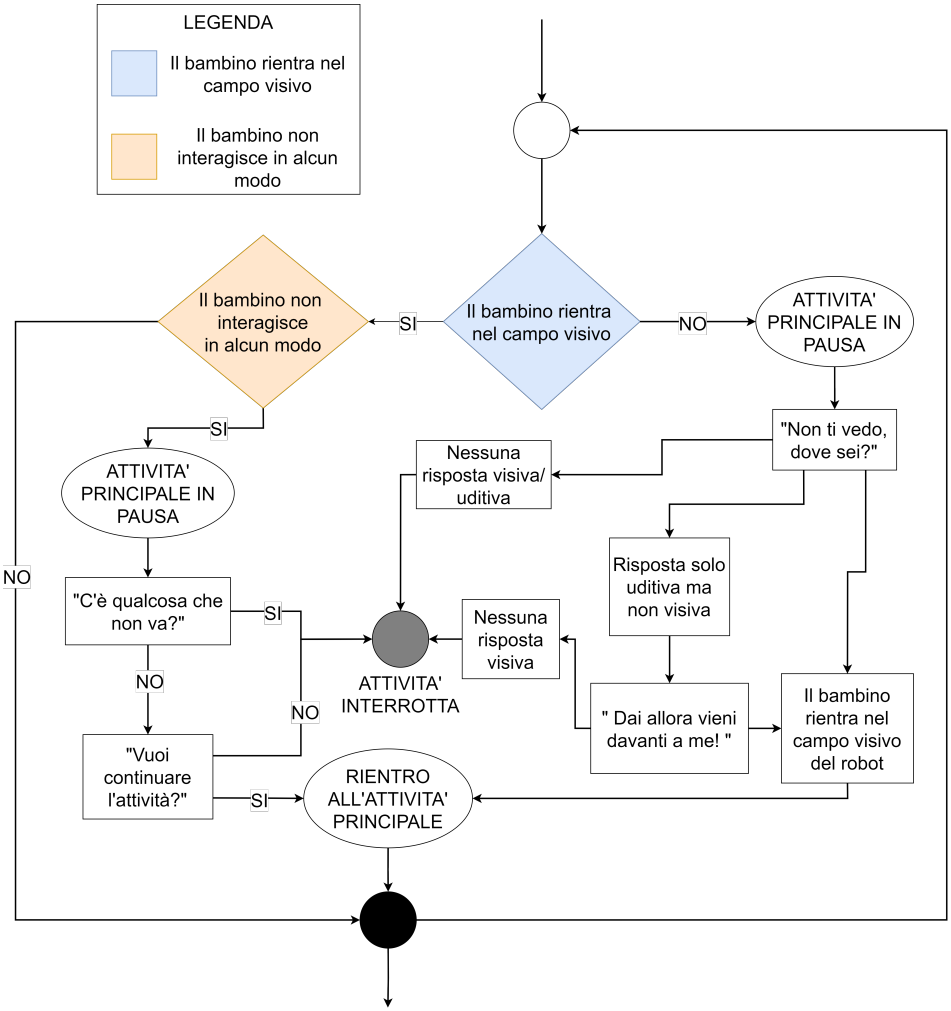


Figure 3.13: This diagram represents the flow of the background's checking, this checks are performed continuously during the activity.

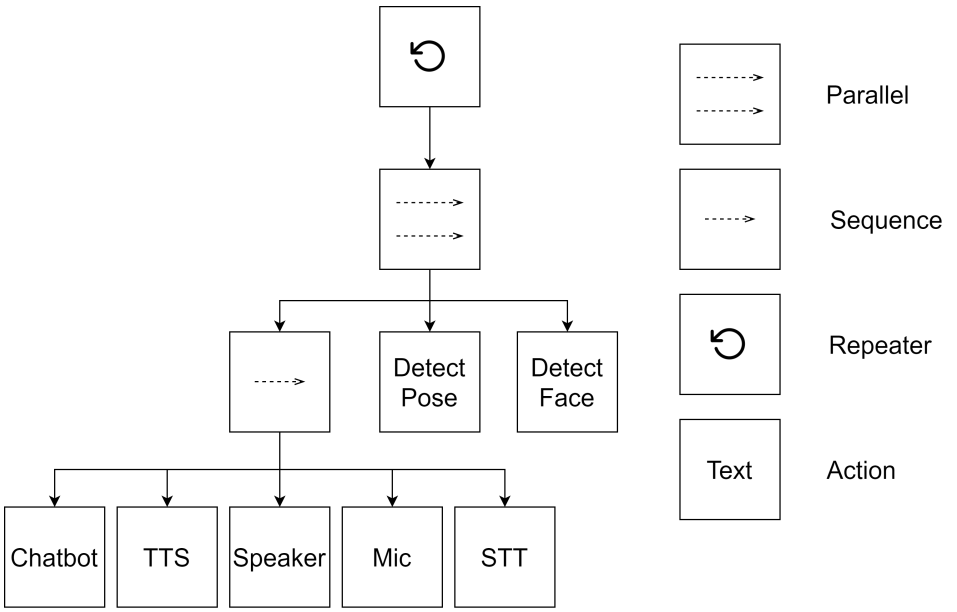


Figure 3.14: Behaviour tree of visual checking that shows the modules needed

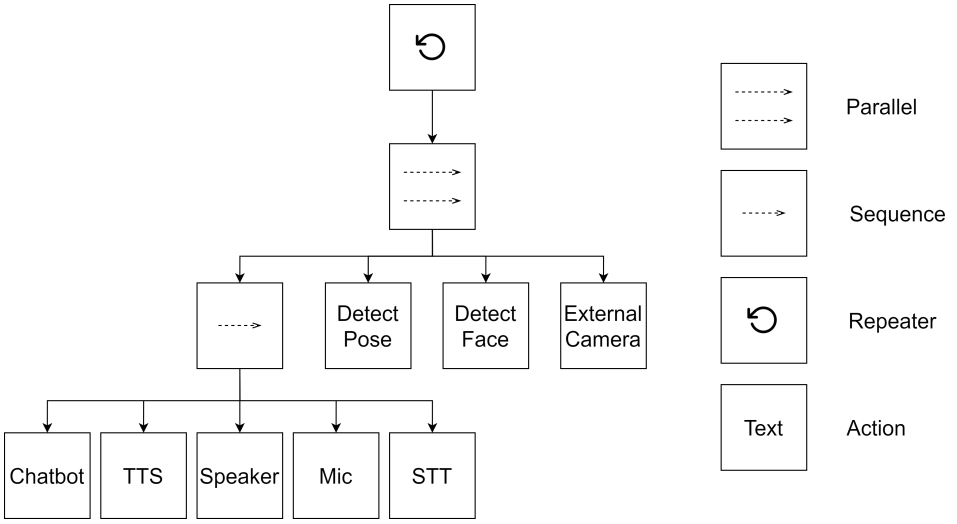


Figure 3.15: Behaviour tree of interaction checking that shows the modules needed

Chapter 4

Implementation

4.1 HARMONI

Human And Robot Modular OpeN Interactions (HARMONI) is a comprehensive tool containing all the components and capabilities developers need to quickly get your social assertive up and running on a robot.
HARMONI packages needed for social human-robot interaction are developed on top of ROS by wrapping several state of the art libraries in the domains of natural language understanding, dialog, object and face detection, and decision trees. HARMONI is like a glue that integrates these different packages on top of ROS in a standardized way so you don't have to do it from scratch.
By keeping to well defined interfaces with loose coupling and encapsulation HARMONI is a modular distributed system. HARMONI is built to be Robot-Agnostic, Modular, and Composable. HARMONI wrap the whole platform in Docker and it works on most hardware platforms and several operating systems [52].
This modularity further allows the system to be created to suit the user's needs just through configuration files.
Now we are going to go in details of HARMONI framework. We remember that HARMONI *was already implemented* and it's not something that we did on our thesis.

4.1.1 Goals

HARMONI is based on three design goals:

- Robot-agnostic
- Modular
- Composable

Robot-agnostic

The HARMONI API is designed to be independent of any particular robot platform. To make that possible, the hardware interface modules are implemented at the lowest layer of the API so that they can be easily replaced without creating dependency problems. Integrating the API with a new robot platform requires only to build modules around hardware endpoints and satisfying the well-defined interface for each type of end point. [52].

Modular

It has been shown that much of the success of ROS and micro-services came through their approach to make packages modular in distributed systems. Taking lessons from these, HARMONI aims to reach similar success with its approach to create modular components while remaining aware of best practices in software engineering. We believe that the design of interfaces is of particular importance in achieving robust modularity. ROS already has excellent interface design but is way to general. We think that flexibility can be overwhelming for new developers and interaction designers so by taking an HRI-centered approach, module interfaces can be more constrained and therefore simplified. Through the simplification of the HARMONI interface, new researchers and developers can learn faster with the added benefit of having simple and clean module interconnection schemes [52].

Composable

Modules should be easily combined and controlled in order to create or extend existing modules into more complex elements. In order to reach this goal, the tool implicitly dictates that module control and error handling follow a hierarchy. This allows the whole HARMONI components to supply very general instructions that are divided as the chain of command is traversed downward. Errors should be handled at the lowest level possible, leaving the top level to deal only with serious or unrecoverable situation. This has basically two benefits: it reduces duplicate modules that typically have fewer contact points, and it decreases the amount of micromanagement code that programmers tend to add in high level modules [52].

4.1.2 Structure

All the ecosystem is a modular tool built around a simple design pattern called the HARMONI Unit. Every module in the ecosystem is a structured pattern and API skeleton is present in HARMONI Unit. The central behaviours needed to run HARMONI are grouped in an ensemble of packages called the HARMONI Core.
These packages manage and orchestrate all the other units and functions of the tool. The rest of the tool is deployed into a modular architecture that supports simple interactions. [52].

HARMONI Unit

The HARMONI Unit employ the ROS action server for handling state and control across the distributed modules that make up the tool. This design pattern, represent and standardizes the communication needed for every element to communicate with the rest modules, including other control units, informational topics, and stored files or parameters. Every unit exposes a fixed API, allowing other units to control it, without the need of have knowledge of its internal structure. This simple API specifies functionality for a given unit to 'do' a certain action,

to ‘request’ a response from an external service, or to ‘start’, ‘stop’, or ‘pause’ a running action.
The difference between ‘do’ and ‘start’ is both in duration of the command (‘start’ is intended for time-extended actions) and in the function arguments that you have to pass (‘do’ typically requires an argument describing what to do, whereas ’start’ should be inherent to the unit). Let show an example: a microphone node can be commanded to ‘start’, ‘stop’, or ‘pause’ listening while a text-to-speech node may receive a string ‘request’ to Google STT and return text speech. In the same way, a motor control node may be commanded to ‘do’ an arm wave or shake [52].

HARMONI Core

HARMONI Core module is responsible of the control and the orchestration of the high level component. It consist of a set of ROS packages including the HARMONI Common Library, the Recorder, and the Pattern and Decision Manager. The HARMONI Common Library incorporate the declaration of the HARMONI Unit and HARMONI Action Server, used by all modules to create a manifest of their functionality to the rest of the tool. This functionalities are guided with a two-level controllers. The first one (lower-level controller) is called harmoni pattern and it is a ROS package, it is used to play any node that follows the HARMONI Unit profile. Here, pattern nodes are grouped together in simple behavior trees to create higher level behaviours. Let’s show an example: to create an interactive spoken dialog, a script should loop the sequence of [chatbot, microphone, speaker, speech-to-text, chatbot, text-to-speech], in which the output of each step is used as input for the following one. These patterns can be created in numerous ways. As an example the visual background pattern is shown in figure 4.1.
Is known that behavior trees are capable of generating super sophisticated behaviors but patterns modalities do not contain logic or the capability to make high-level decisions. This is the reason why we added pytree in HARMONI ecosystem. Decision Manager is the unit that is in charge of directing the flow of patterns and handling decision making. The Decision Manager operates as a higher-level controller that

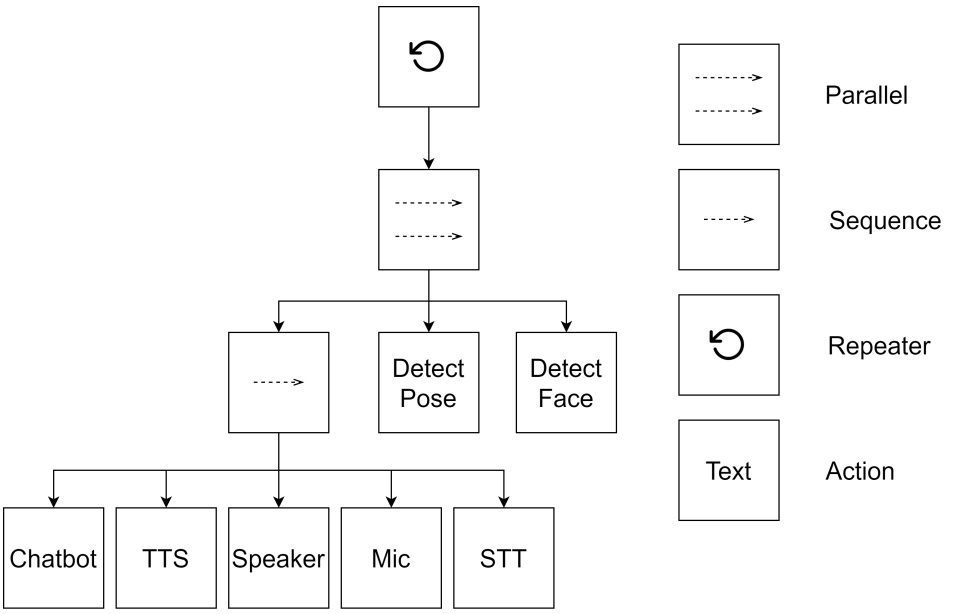


Figure 4.1: A sample HARMONI behavior, structured as a behavior tree that starts robot speech sequence, followed by pose and face detection. Each leaf node is an API call to the specified HARMONI Unit [52]

looks for the state of the system and receives feedback from all the patterns in order to make conscious decision and choose what pattern should be played next. The Decision Manager can be implemented in several ways like as a decision tree, behavior tree, or custom script, depending on the length and needs of the interaction, before our work it was implemented as a custom script but now it is implemented as a behaviour tree. The Decision Manager starts up nodes and handles any errors that arise. Last, but not the leas, in order to help the Decision Manager with memory for long-term or repeated interactions, we supply the Recorder Package, which Manager to store information in a database that can be queried [52].

System Integration and Architecture

The HARMONI Unit and Core, described above, make up the skeleton of the tool. They give order and command to modules in order to enable communication between all the units.
Any existing functionality for example, a piece of hardware, or cloud

services, or a software, is easy to be added in HARMONI with a simple Python wrapper.
This structure enables the largest HARMONI contribution, the systematic integration of disparate functionalities is easy and straightforward and so it's easy to conduct a meaningful social human-robot interaction. So as to manage the multiple capabilities and different configurations that could be desirable for a social human-robot interaction, HARMONI adopts a standard hybrid sense-plan-act architecture.
As can be easily seen in this communication diagram, the *Decision Manager* directly controls the *Pattern Player* and check for status of each pattern that come in play. The Pattern Player can directly command instruction to each HARMONI Unit, since it is an interface and a controller for the hardware or service it exposes. Units can communicate directly with each other through a continuously streaming of data. This communication is easily done thanks to ROS publisher and subscriber.
In this architecture, new capabilities are readily added and the only things that developers must do is follow the simple HARMONI Unit convention.
In figure 4.2 is shown HARMONI architecture.
Thanks to this organization, HARMONI has a lot of flexibility in the management and definition of modules that must play in a specific and ordinary activity. Despite that there is no way to add behaviours like the ones that you can have with the so called *behaviour trees* [52].
So the fist part of our thesis work was on the integration of Behaviour Trees in HARMONI ecosystem.
There were many ways of doing behaviour tree. In literature you can find some behaviour tree engine like [53]:

- *Behavior Tree*: Behavior Tree library in C++
- *py_trees* : Behavior Tree library in Python
- *Groot*: Graphical Editor to create BehaviorTrees
- *CoSTAR*: Stack Collaborative System for Task Automation and Recognition

But we decided to proceed with PyTree since it is created with Python programming language, it is widely used and it already has been prepared to be used with ROS.
The following section with goes in details about PyTree and the steps that we maid to import that package in HARMONI. [54]

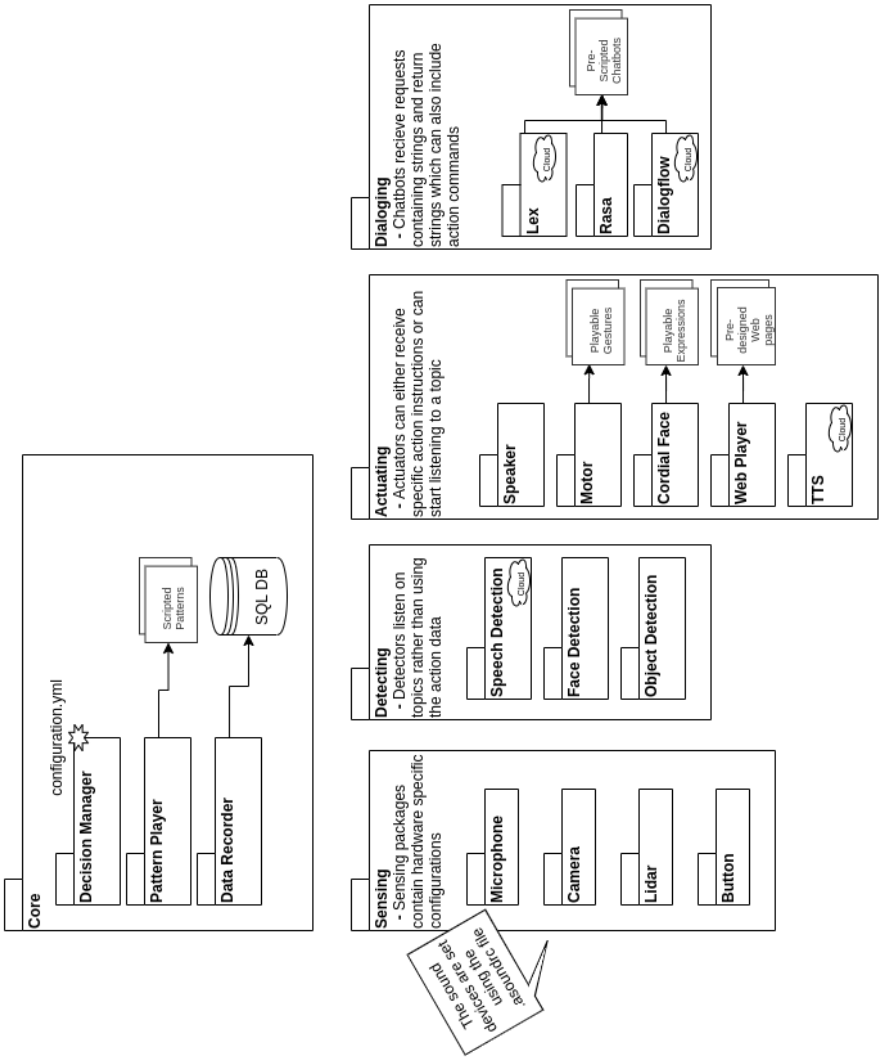


Figure 4.2: This figure shows HARMONI architecture. Decision Manager and Pattern Player in the core unit were used before behaviour trees implementation [52]

4.2 PyTree

A great part of our thesis work was done on PyTree and its integration in HARMONI ecosystem.
The need of PyTree comes from a deeper concept which was the need of adding **behaviour trees** in HARMONI. Indeed, PyTree is one among the several ways to implement behaviour trees.
A behavior tree is a mathematical model of plan execution used in computer science, robotics, control systems and video games. They describe switching between a finite set of tasks in a modular fashion. It is graphically represented as a directed tree in which the nodes are classified as root, control flow nodes, or execution nodes (tasks). For each pair of connected nodes the outgoing node is called parent and the incoming node is called child. The root has no parents and exactly one child, the control flow nodes have one parent and at least one child, and the execution nodes have one parent and no children. Graphically, the children of a control flow node are placed below it, ordered from left to right [6].
The execution of a behavior tree starts from the root which sends ticks with a certain frequency to its child. A tick is an enabling signal that allows the execution of a child. When the execution of a node in the behavior tree is allowed, it returns to the parent a status RUNNING if its execution has not finished yet, SUCCESS if it has achieved its goal, or FAILURE otherwise [6].
Behavior trees are visually intuitive and easy to design, test, and debug, and provide more modularity, scalability, and reusability than other behavior creation methods.
PyTree is just a Python implementation of behaviour trees, therefore very useful for our purpose. A tree in PyTree is made of two elements: nodes and leaves.
Nodes can be of different nature but in general they belong to the family of composites (sequence, parallel, selector). The management of these elements it's not something that users have to do.
Leaves come from a single element called "behaviour" and these are the components that users can create.
Let's see in more details composites (figure 4.3):

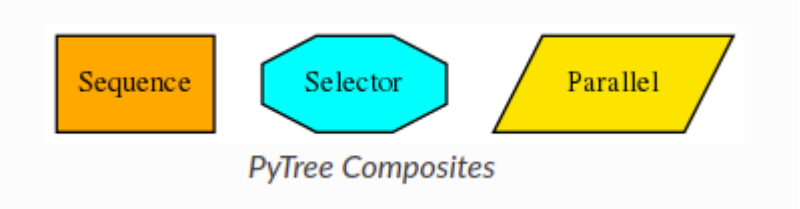


Figure 4.3: This figure shows PyTree Composites [55]. Any activity can be designed and implemented by using a combination of these three elements.

- **Sequence** (figure 4.4): A sequence will progressively tick over each of its children so long as each child returns SUCCESS. If any child returns FAILURE or RUNNING the sequence will halt and the parent will adopt the result of this child. If it reaches the last child, it returns with that result regardless[55]. One should use this composite when there are some tasks that have to be done in sequence one after the other and maybe the output of one component is used as input of the following one.

- **Parallel** (figure 4.5): Ticks every child every time the parallel is run. It will return FAILURE if any child returns FAILURE, and then SUCCESS accordingly to the policy that is uses.
  With policy Success-On-All only returns SUCCESS if all children return SUCCESS.
  With policy Success-On-One return SUCCESS if at least one child returns SUCCESS and others are RUNNING.
  With policy Success-On-Selected only returns SUCCESS if a specified subset of children return SUCCESS [55].
  One should use this composite when there are tasks that are independent one from each other.

- **Selector** (figure 4.6): A selector executes each of its child behaviours in turn until one of them succeeds (at which point it itself returns RUNNING or SUCCESS, or it runs out of children at which point it itself returns FAILURE. We usually refer to selecting children as a means of choosing between priorities. Each child and its subtree represent a decreasingly lower priority

path[55]. One should use this composite where there are different level of priority between tasks and one should be executed only after the other one are finished. In addition if some priority task has to be executed it preempt the execution from tasks that have low priority.

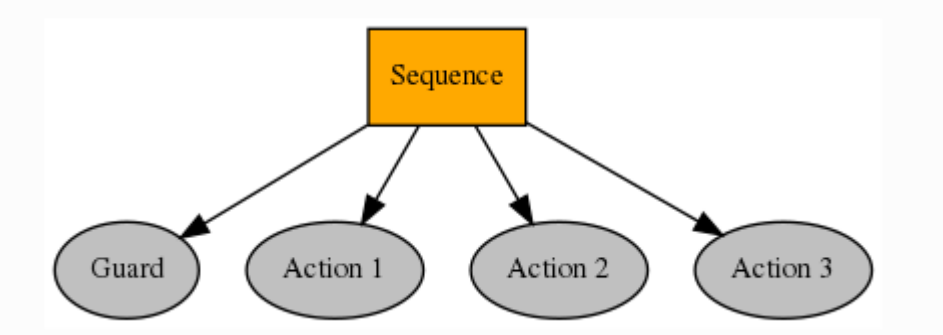


Figure 4.4: Graphical representation of Sequence composite [55]. In these figure the sequence component (orange one) has 4 children.

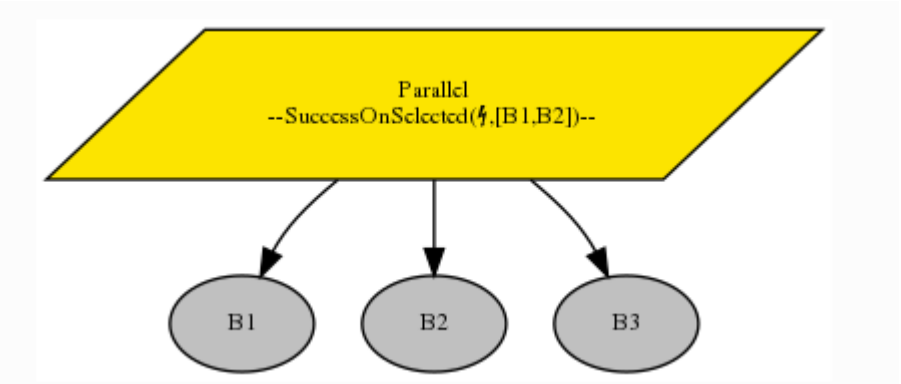


Figure 4.5: Graphical representation of Parallel composite [55]. In these figure the parallel component (yellow one) has 3 children.

PyTree made also several extra elements that developers can use combined to composites to have more complex functions. We want to report here some of them:

- **Blackboards**: Blackboards are not a necessary component of behaviour tree implementations, but are nonetheless, a fairly common mechanism for sharing data between behaviours in the

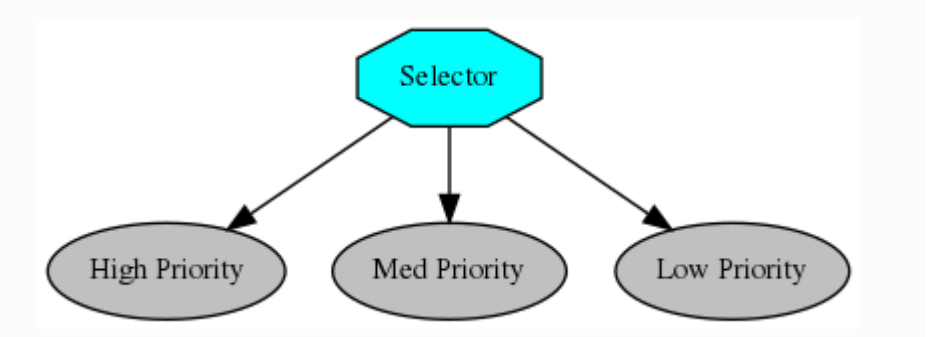


Figure 4.6: Graphical representation of Selector composite [55]. In these figure the selector component (light blue one) has 3 children.

tree. The current implementation adopts a strategy similar to that of a file-system. Each client (subsequently behaviour) registers itself for read/write access to keys on the blackboard. This is less to do with permissions and more to do with tracking users of keys on the blackboard - extremely helpful with debugging[55].

- **Decorator** (figure 4.7): Decorators are behaviours that manage a single child and provide common modifications to their underlying child behaviour (e.g. inverting the result). That is, they provide a means for behaviours to wear different 'hats' and this expands the capabilities of your behaviour library[55].

- **Idioms**: A library of subtree creators that build complex patterns of behaviours representing common behaviour tree idioms. Common decision making patterns can often be realised using a specific combination of fundamental behaviours and the blackboard[55].

Idioms family has a lot of components but one of the most useful for us was the **Either Or**. We want to explain it briefly since it will appear often when we will talk about the body of the tree.
EitherOr (Figure 4.8) is used when you need a kind of selector that does not implement prioritisation, i.e. you would like different paths to be selected on a first-come, first-served basis. Up front is an XOR conditional check which locks in the result on the blackboard under the specified namespace. Locking the result in permits the conditional

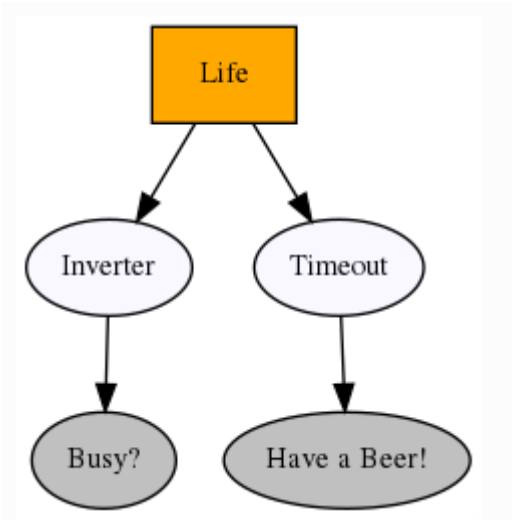


Figure 4.7: Graphical representation of decorator (white component) [55]. This figure shows a sequence (*Life*) with 2 children. The first one (*Busy?*) has an *Inverter* decorator so its states FAILURE and SUCCESS will be flipped before been transmitted up in the tree. The second child (*Have a Beer!*) has a *Timeout* decorator so after a define time its state will pass to SUCCESS no matter what *Have a Beer!* do.

variables to vary in future ticks without interrupting the execution of the chosen subtree (an example of a conditional variable may be one that has registered joystick button presses).
Once the result is locked in, the relevant subtree is activated beneath the selector. The children of the selector are, from left to right, not in any order of priority since the previous xor choice has been locked in and it is not revisited until the subtree executes to completion. Only one may be active and it cannot be interrupted by the others.
The only means of interrupting the execution is via a higher priority in the tree that this idiom is embedded in.

In the following paragraph we describe the **Behaviour** which is the principal component in PyTree world: every leaf in tree must be a behaviour. Behaviours are usually representative of either a check or an action and they are the principal components that works with blackboards.
Since a leaf must extend the Behaviour class let's see more in detail which are the function that developers must define and what they do:

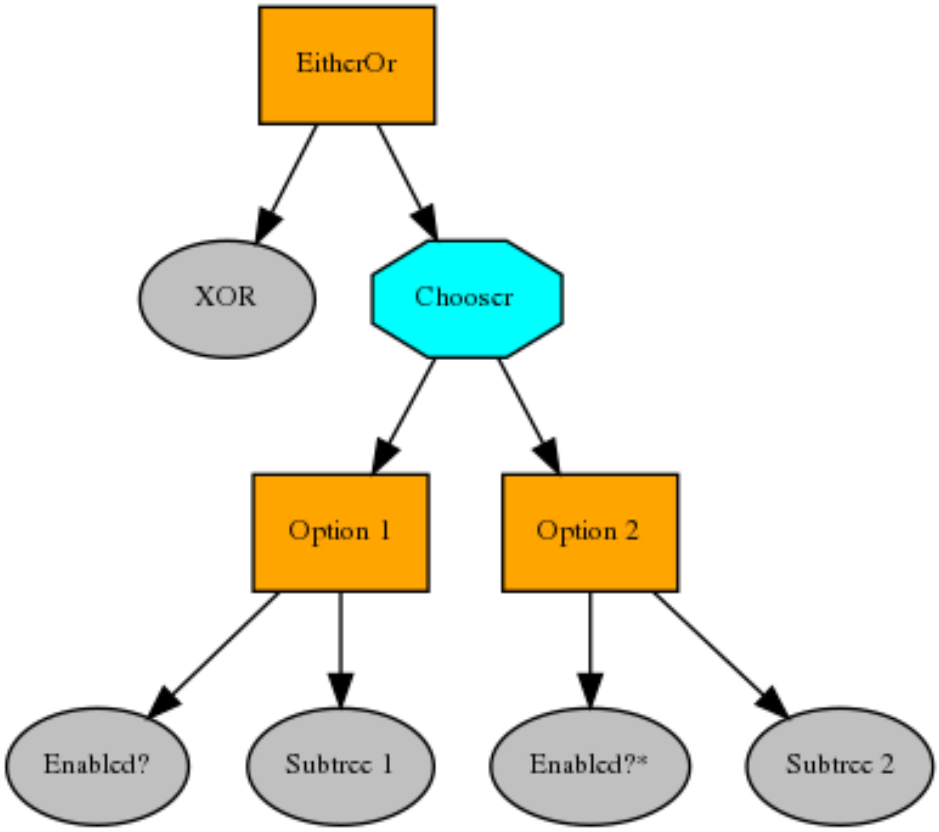


Figure 4.8: This figure show an EitherOr component [55]. This component will check for two boolean conditions and will run either *Subtree1* or *Subtree2*.

- `__init__`: here goes a minimal one-time initialisation. A good rule of thumb is to only include the initialisation relevant for being able to insert this behaviour in a tree for offline rendering to dot graphs.
  This method is called one time when the object is created [55].

- **`setup`**: here goes delayed one-time initialisation that would otherwise interfere with offline rendering of this behaviour in a tree to dot graph or validation of the behaviour's configuration.
  Also this function is called one time but it should be either manually called by this behaviour alone, or more commonly, by the tree [55].

- **`initialise`**: here goes any initialisation you need before putting your behaviour to work.
  This method is called the first time your behaviour is ticked and anytime the status is not RUNNING thereafter[55].

- **`update`**: as the documentation suggest [55], in this function you should *triggering, checking, monitoring. Anything...but do not block!*. Simply put, here goes the core of your leaf. One should also set a feedback message and, last but not least, return one of the PyTree status (RUNNING, SUCCESS, FAILURE).
  This method is called every time your behaviour is ticked [55].

- **`terminate`**: here goes everything that needs to be done after the leaf is done working or, when a priority part of the tree needs to work.
  This function is called whenever your behaviour switches to a non-running state. (SUCCESS, INVALID, FAILURE) [55].

In Appendix A (page 149) you can find the documentation of PyTree that we wrote for all HARMONI 's user.

4.3 Integration

In this section we will present the steps that we did in order to create the behavioural tree of the whole activity. Our work can be divided in

two part:

- Design and implementation of the *body* of the tree.
- Implementation of all the *leaves* (Behaviours) thought in the design.

As we said before, HARMONI works as a client server architecture. In some cases (for example ImageAI or Google Speech-To-Text) the HARMONI servers weren't implemented, so we had to create them.
Now we will present the tree of our activity and all the leaves that we create.

4.4 Body

The steps that bring to the construction of the body of the tree are quite a lot. We went through three phases such as design, implementation and test. We reiterated these three phases two times: the first one was for the creation of the skeleton without consider leaves, the second one was done after the creation of the leaves and it was an adjustment phase.
For all these stages we decided to apply a "top-down" approach so we define first a big picture of the tree and then we went in details.
In this section we will explain just the structure of the tree, Behaviours (leaf) will be explained in the following section in a clear way.
The root of the tree was realized with a *Selector*. We recall that selector is useful when you have to define levels of priority between children and this is the case: the first child is the root of the entire activity, the second child is the root of the *reset* module while the third child is just a *Running Behaviour leaf* of PyTree. In general *Running Behaviour* is used to force "idle" functioning while a *Selector* is used.
This part of the tree is shown in figure 4.9.

We can notice that the first child of the root is a Parallel composite and it has two children. The first one is called *AdLibitum*; this name was not chosen by chance in fact it refers to some modules that have to work at all times till the *Session* module is working.

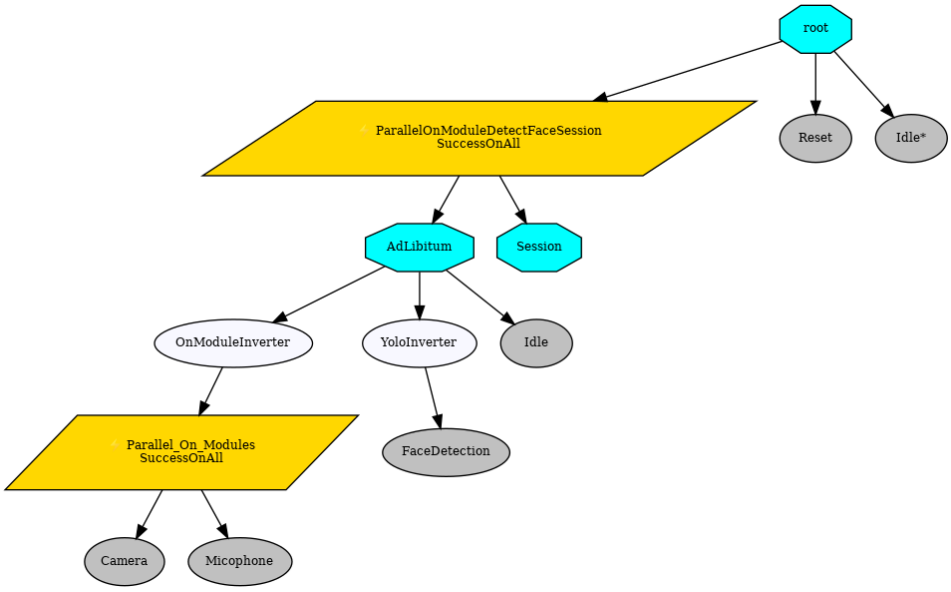


Figure 4.9: This figure shows a macro of the whole tree. In the first level there are 3 children, the first one is a parallel composite that in the head of the all activity. The second child is the component in charge of reset all the modules when the activity end. The third component is just an *Idle* behaviour and it is used to block the tree after *Reset* behavior worked. The first parallel composite has 2 children, the first one is a selector that is used to control sensor components. The second child called Session composite will be shown in detail later.

The *Session* is the root of the actual activity, we will explain it later. Coming back on AdLibitum node we can see that there are three children, the last one is again a *Running Behaviour leaf*, while the other two are used respectively for sensor modules and for the detection of the person.
White component are Inverter, family of *Decorators*, and they are used just because Selector goes ahead in ticking next child only if the current child return FAILURE as state. Our Behaviours return SUCCESS if the results are correctly and so we need an element that was able to invert the state form SUCCESS to FAILURE and vice versa.
Camera and *Microphone* leaves manage the homonyms sensors devices while *FaceDetection* is the leaf that control plain yolo service.
Let's concentrate now on *Session* node shown in figure 4.10.
This Selector has seven children, four of them are Sequence composites while the other three are Inverter. The first child is the Therapist node, this node controls the "call therapist" event, every time that the tree decide that the therapist is need this module take action. In figure 4.11 there is a zoom in this module and readers can see that this is just a blocking behavior like the *Idle* node in the first level of the all tree (figure 4.9). Once can notice the similarity between this subtree and the *EitherOr* component in figure 4.8
Apart form the first child, let's consider these children grouped two at a time; indeed these groups are used to better control the flow of the activity. Inverter are placed on top of backgrounds and main-activity, the reason is always the same as before: invert the meaning of the return state of these subtrees. All the other sequence are special subtree called *Running Or Failure*; these subtrees are just control nodes and are used to understand, form the Session composite prospective, what happened in the backgrounds and in the main activity.
To recall what backgrounds and main activity are, we suggest having a look on 3.7.1 and 3.7.2 in pages 29 and 31.
It's important to remember that the children of a selector are placed in priority order, so therapist is the first child that is ticked, then visual background and so on. In the creation of the tree we used a lot of *EitherOr* component of *Idioms* family of PyTree.
We will now show theses three subtrees in details.

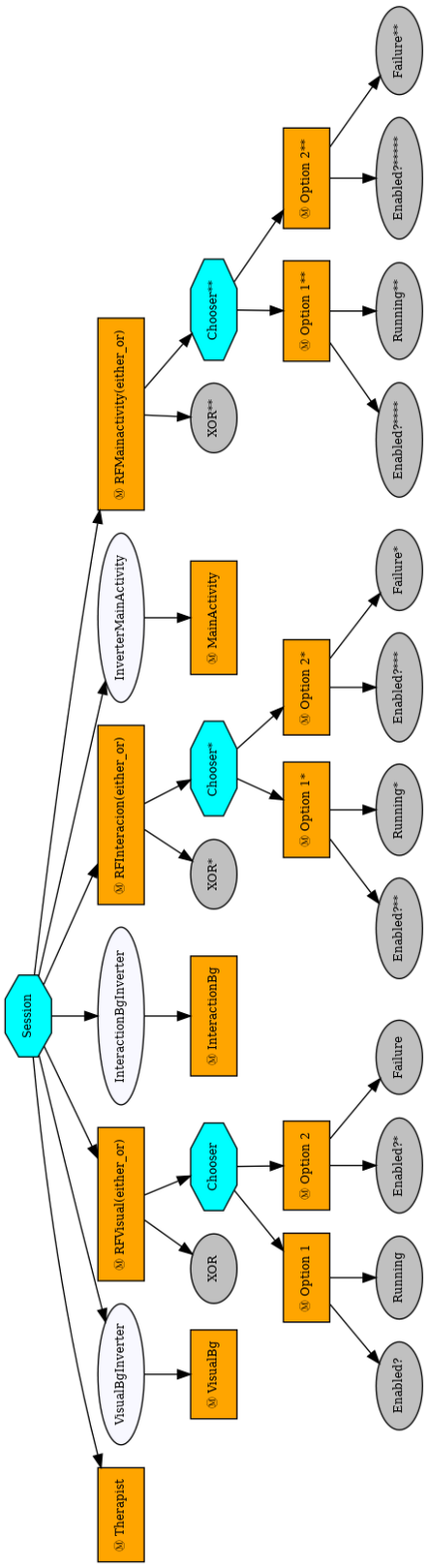


Figure 4.10: This figure shows the Session node in details (seen for the first time in figure 4.9). This selector is the most important node for the activity since it controls the execution flow of the two backgrounds, of the main activity and of the therapist intervention. It is in charge of change the execution between modules accordingly to the answers and actions of the user.

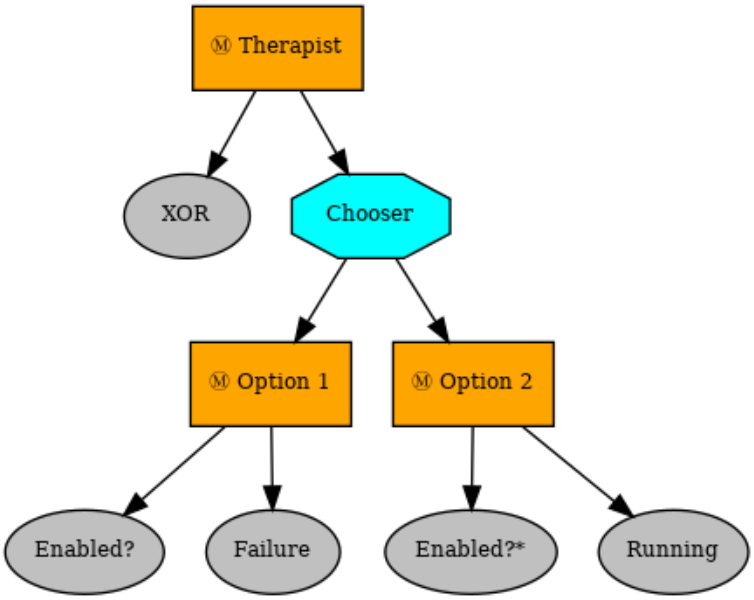


Figure 4.11: This figure shows the Therapist subtree. This part of the tree is executed every time the therapist should come in help. This subtree stops the activity.

Figure 4.12 shows visual background subtree, this node has two children, the first one is an EitherOr while the second one is a special Behaviour used to control the scene of this subtree, we will clearly explain all the Behaviours in the next section.
The EitherOr is use just to direct the ticking of the tree in the right direction, it decide whether or not execute this background based on the result of the *FaceDetection* behaviour.
Let's suppose to enter in this background (we are in *SequenceVisual* node of figure 4.12). Simply put, we can dived this (sub)subtree in two part: robot driven and child driven (this is a general concept that is valid for all the backgrounds and for the main activity).
The first part include all leaves that are used to control the robot such as face, speech and so on; the second part include all the leaves that are used to grasp answer and action of the user that is playing the activity.
Regarding the robot side one important aspect is the one related to the *AwsLexTrigger* leaf. Here the question is the following, do we have to trigger amazon lex to make a question or we are just saying a sen-

tence that don't expect an answer? The EitherOrTrigger answer this question and chooses between one or the other path.
This is a special background since it is supposed to run when the user is not detected in front of the robot so, the part related to the user answer is traduced in only the FaceDetection behaviour.
In figure 4.13 you can find the subtree of the interaction background. The structure is similar to the one of the visual, in the first level there is one EitherOr, used to run this background if it is the case, and the subtree result (like in the visual).
Here we can appreciate more the difference between the robot part and the user part in the children of the *SequenceInteracion* node. Indeed, the *EitherOrKid* node is the one that control the Kid answers.
The last subtree is the Main Activity one and it is shown in figure 4.14. Again the structure of this subtree is similar to the one of the two backgrounds but in this part all the Behaviours come in play. In the first level we can see three children that are: *SequenceRobot*, *ParallelFaceGestureKid* and *SubtreeMain*.
The first one manages the robot, the second manages the kid, facial expression and gesture, and the latter is used, as always, for internal purposes. Here there is no EitherOr on top of the tree because if we have reached this point of the tree we are sure that *Main Activity* must play.
We can schematize the steps of the tree like that:

1. Setting up web page to be printed, sound and sentence of the robot,
2. Conversion of the sentence in audio file and reproduction of the latter combine to lips synchronization,
3. Recording of the voice, cards and buttons (if any) for the answer of the user,
4. Analysis of the results combine to some pre-computation steps,
5. Repetition from 1. until the end of the story is met.

The whole activity is represented in figures 4.15, 4.16, 4.17, 4.18, 4.19.

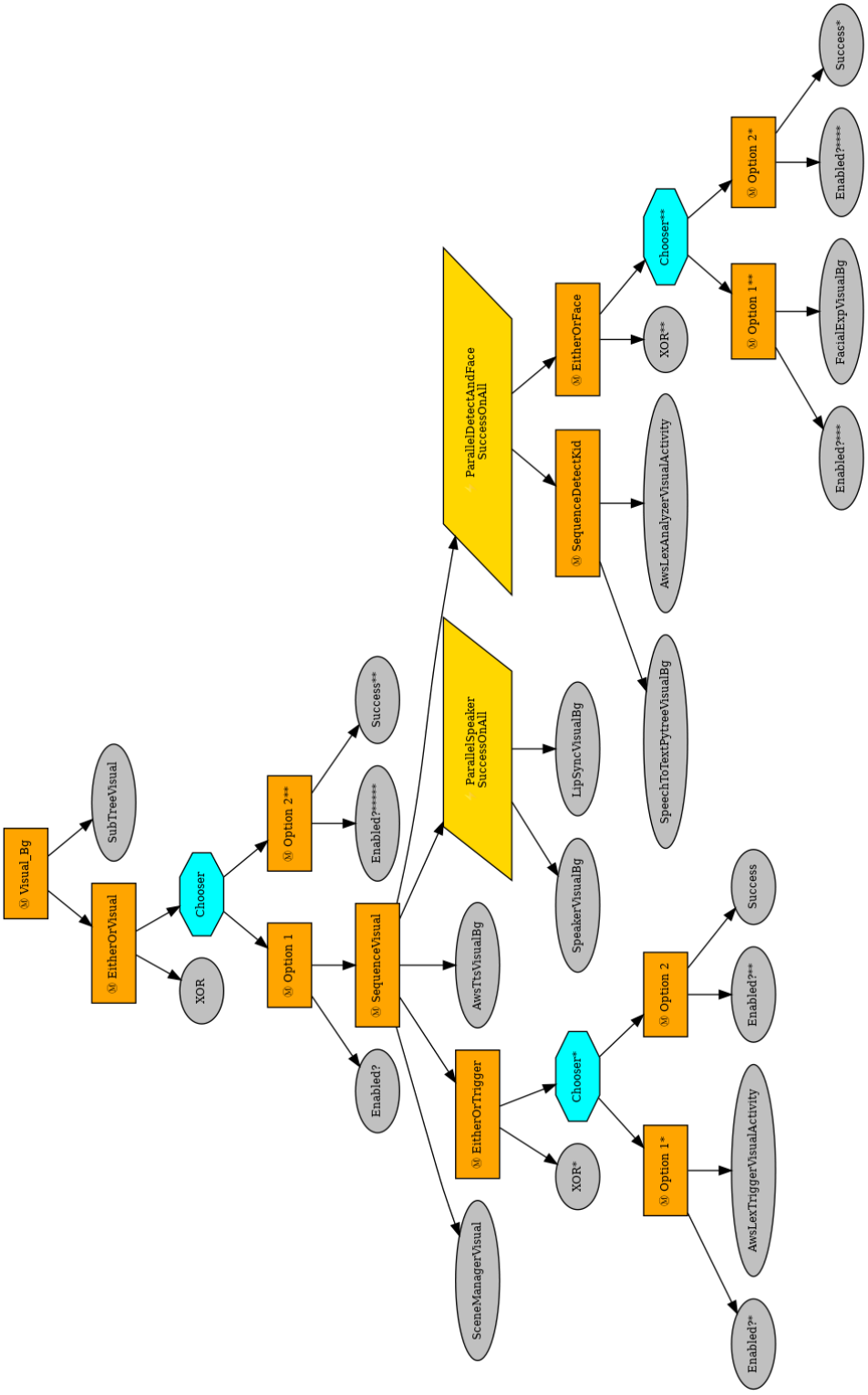


Figure 4.12: This figure shows the Visual Background subtree. This child takes care of the visual background that is running while the used is not seen by the face detection module.

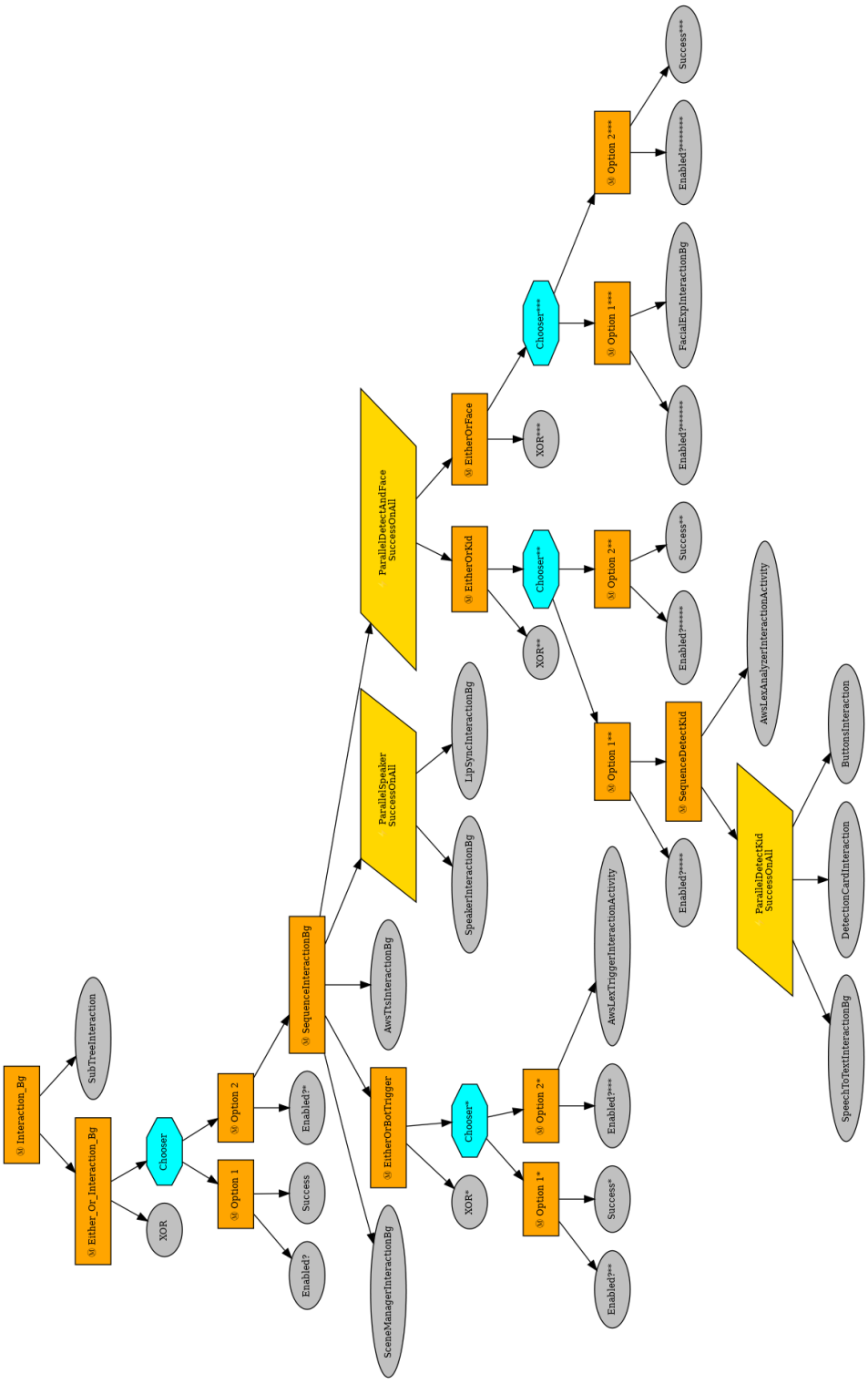


Figure 4.13: This figure shows the Interaction Background subtree. This child takes care of the interaction background that is running when the user doesn't interact with the robot.

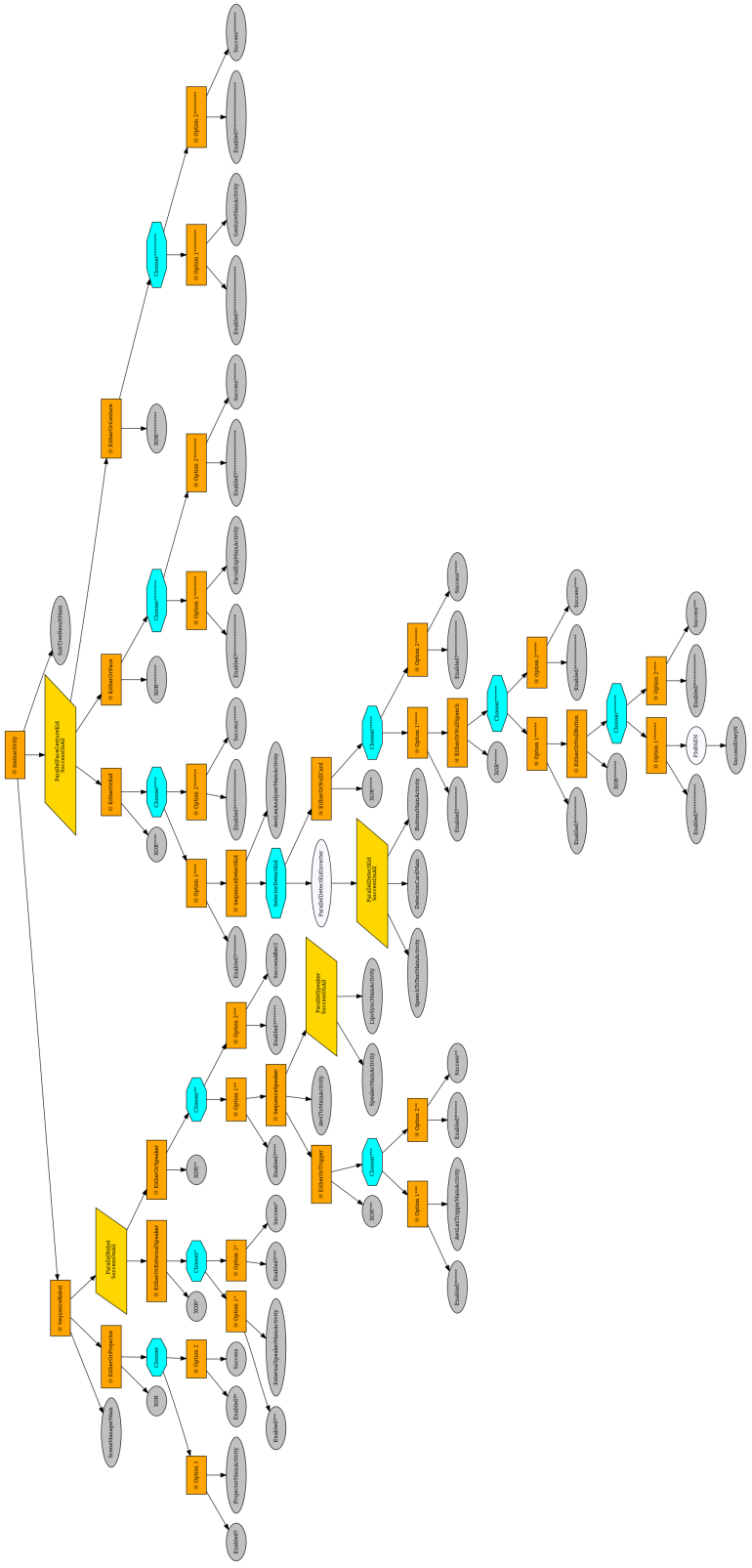


Figure 4.14: This figure shows the Main Activity subtree, this is the core of the activity. This child is in charge of running the recycling part of the activity.

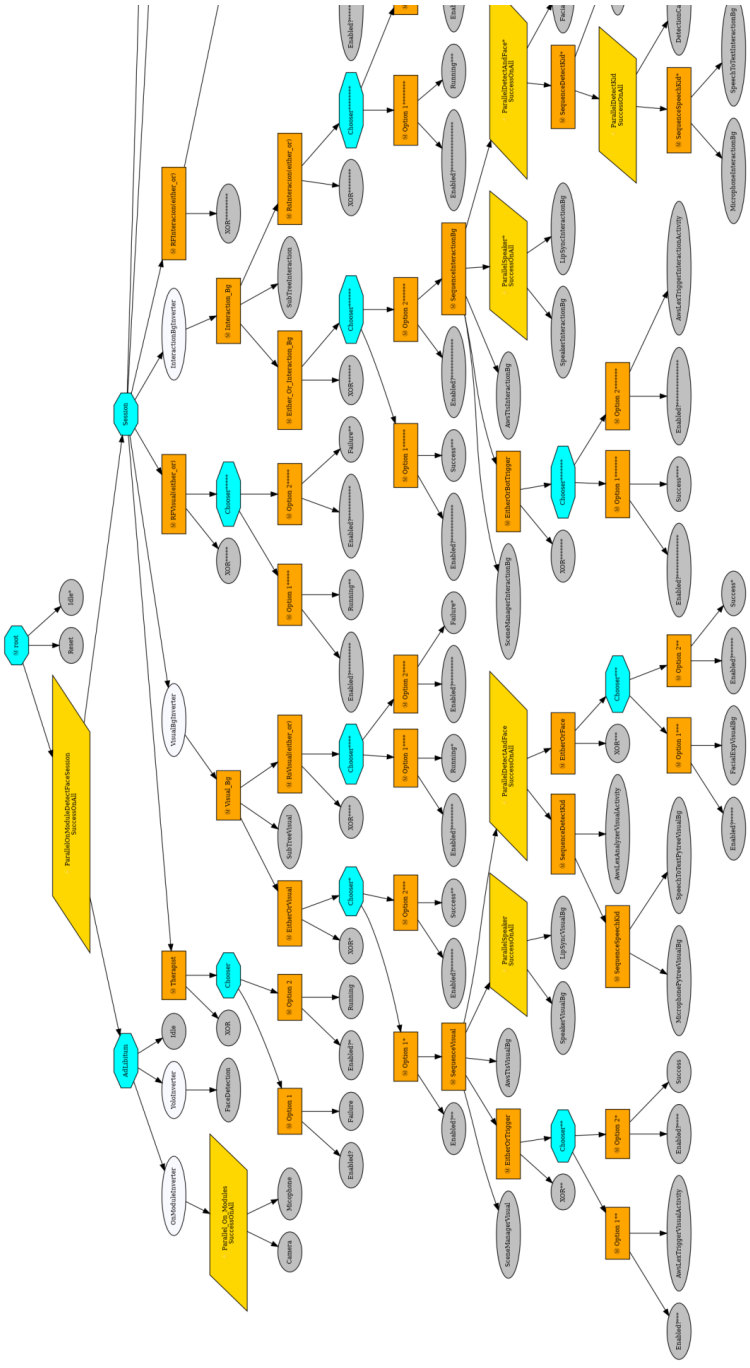


Figure 4.15: The is a zoom of the entire tree of the activity.

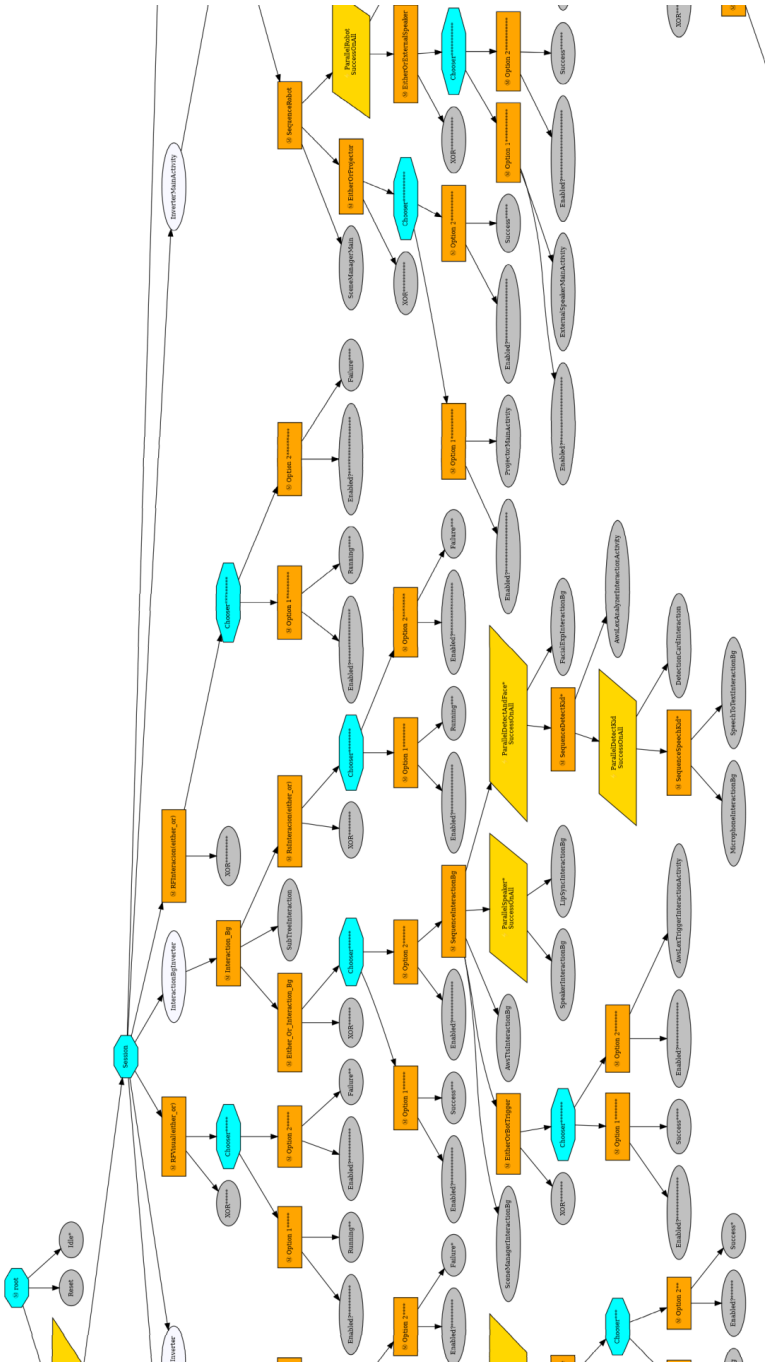


Figure 4.16: The is a zoom of the entire tree of the activity.

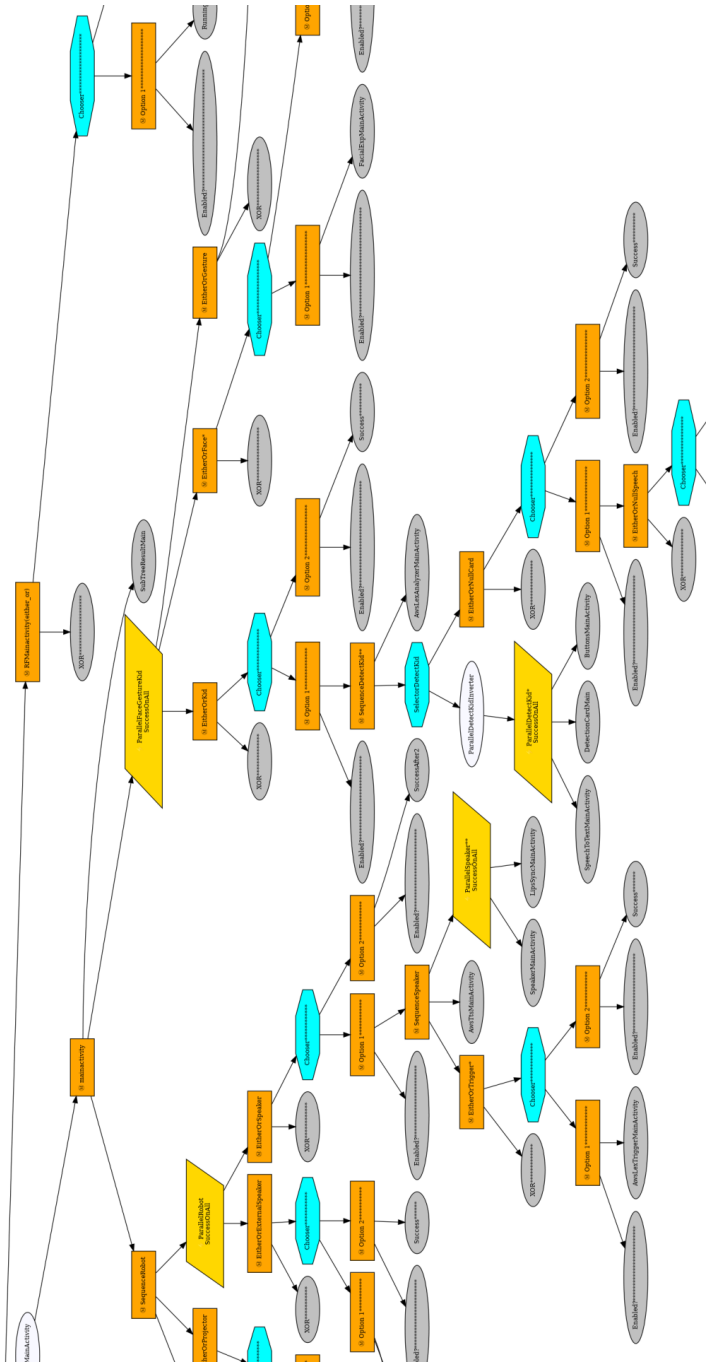


Figure 4.17: The is a zoom of the entire tree of the activity.

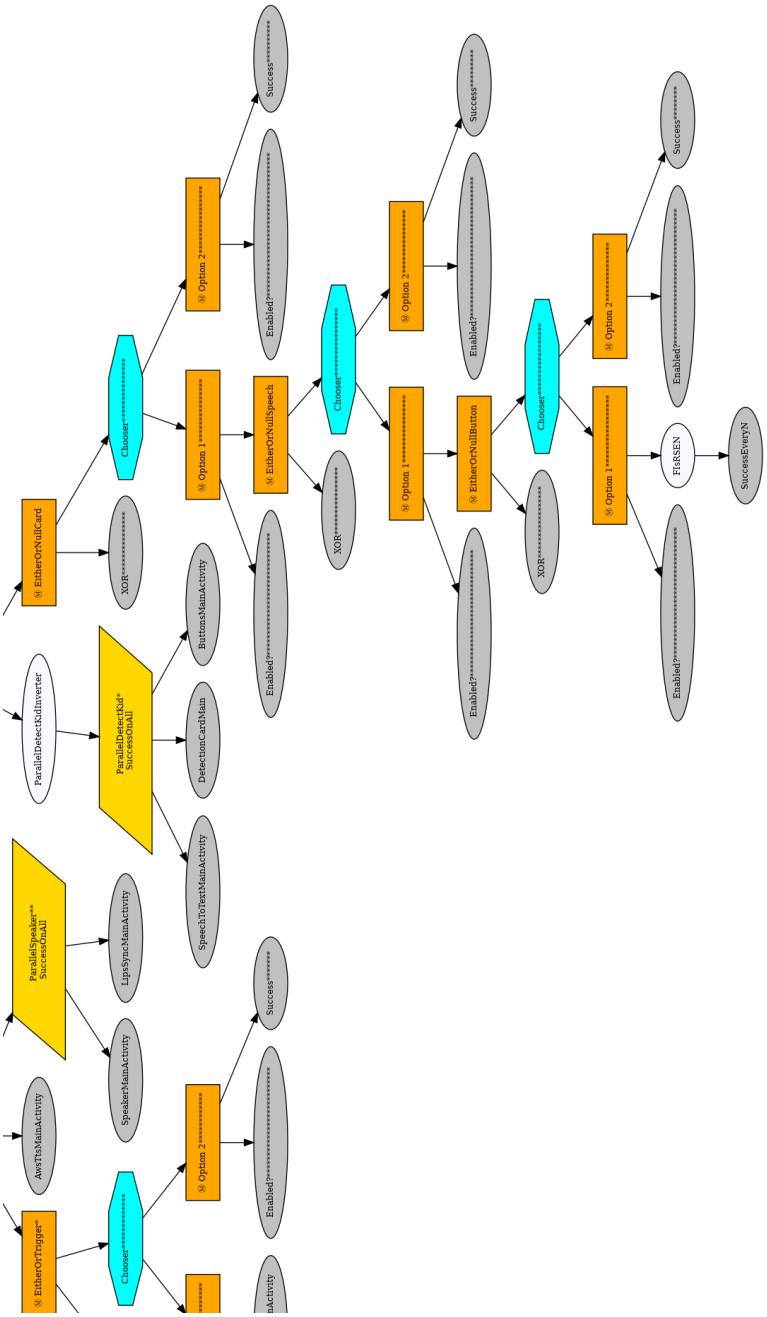


Figure 4.18: The is a zoom of the entire tree of the activity.

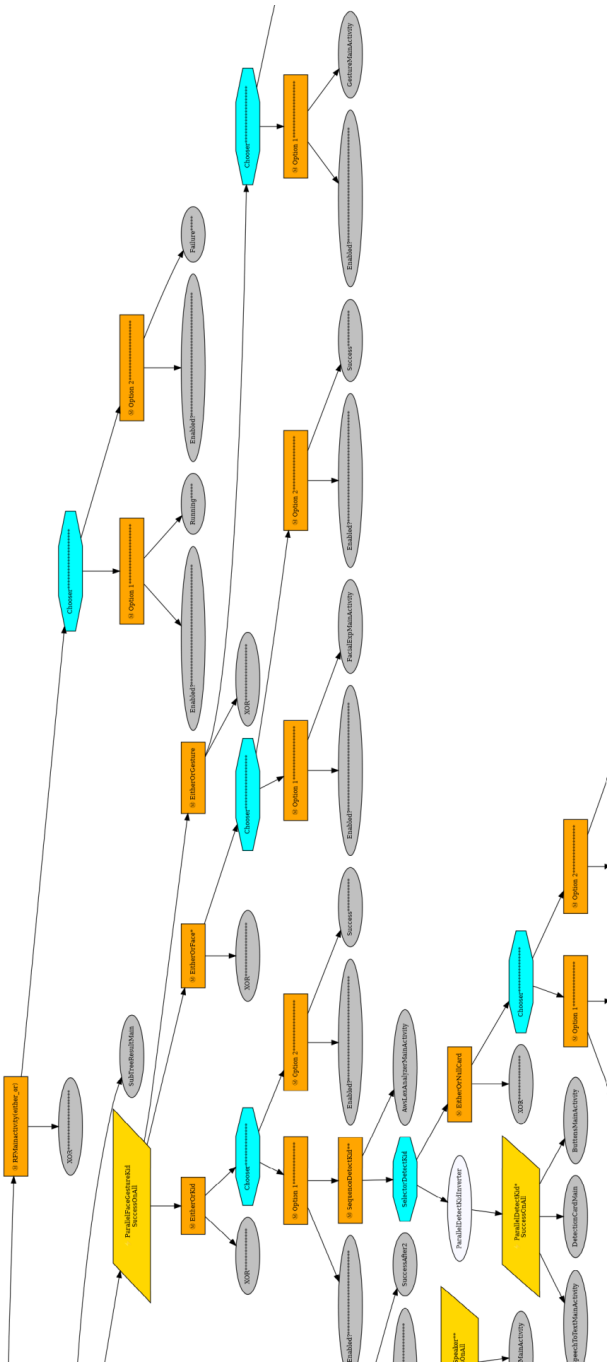


Figure 4.19: The is a zoom of the entire tree of the activity.

```
class PyTreeNameSpace(Enum):
    scene = "scene"
    visual = "visual"
    interaction = "interaction"
    mainactivity = "mainactivity"
    timer = "timer"
    invalid_response = "invalid_response"
    analyzer = "analyzer"
    trigger = "trigger"
    buttons = "buttons"
```

Figure 4.20: This figure shows PyTree namespaces for blackboards that we create for HARMONI framework.

The last, but not the least, thing that we want to report here are the namespaces that we create for blackboards, figure 4.20.

4.5 Leaves

In the following subsections we will explain one by one all the leaves of the tree in details.

4.5.1 ImageAI

Brief introduction

Nowadays object detection is implemented thanks to neural networks; so as to create our object detection component we needed a solution based on machine learning. After several studies we found what we thought would be a great solution of our scopes: ImageAI[56].
ImageAI is a python library built to empower developers to build applications and systems with self-contained Deep Learning and Computer Vision capabilities. Built with simplicity in mind, ImageAI supports a list of state-of-the-art Machine Learning algorithms for image prediction, custom image prediction, object detection, video detection, video

object tracking and image predictions trainings.
ImageAI currently supports image prediction and training using 4 different Machine Learning algorithms trained on the ImageNet-1000 dataset. It also supports object detection, video detection and object tracking using RetinaNet, YoloV3 and TinyYoloV3 trained on COCO dataset. Finally, ImageAI allows users to train custom models for performing detection and recognition of new objects. We decided to use ImageAI since it has a great documentation and because it provide a simple and effective methodologies to create your own custom models. Create a neural network for object detection from scratch is an hard work so we decided to find the state of the art in that field and use it.

Implementation in HARMONI

For our purposes we needed two different models:

- One for recognizing person.
- One for recognizing a set of cards.

Regarding the first model we decided to use the already working yolo.h5 create for ImageAI team. This model is the one trained on ImageNet-1000 dataset. So as to use this models we just had to download the already existing model and then feed it to the neural network.
Since the fact that this model was created by ImageAI teams the performance are astonishing. The model is able to recognize person in many different position and also when the person is semi-obfuscated by object or semi-cutted by the camera frame.
The neural network model is called YoloV3 and in figure 4.21 there are all the layers of while in figure 4.22 (page 73) there is the architecture [57].

Layer	Filters size	Repeat	Output size
Image			416 × 416
Conv	32 3 × 3/1	1	416 × 416
Conv	64 3 × 3/2	1	208 × 208
Conv Conv Residual	32 1 × 1/1 64 3 × 3/1	Conv Conv Residual × 1	208 × 208 208 × 208 208 × 208
Conv	128 3 × 3/2	1	104 × 104
Conv Conv Residual	64 1 × 1/1 128 3 × 3/1	Conv Conv Residual × 2	104 × 104 104 × 104 104 × 104
Conv	256 3 × 3/2	1	52 × 52
Conv Conv Residual	128 1 × 1/1 256 3 × 3/1	Conv Conv Residual × 8	52 × 52 52 × 52 52 × 52
Conv	512 3 × 3/2	1	26 × 26
Conv Conv Residual	256 1 × 1/1 512 3 × 3/1	Conv Conv Residual × 8	26 × 26 26 × 26 26 × 26
Conv	1024 3 × 3/2	1	13 × 13
Conv Conv Residual	512 1 × 1/1 1024 3 × 3/1	Conv Conv Residual × 4	13 × 13 13 × 13 13 × 13

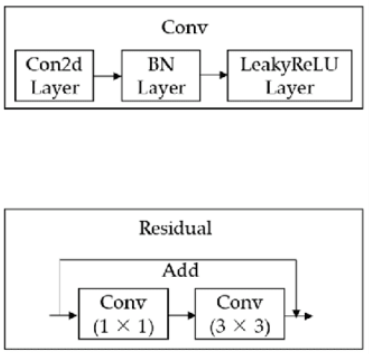


Figure 4.21: This figure shows YoloV3 layers [57].

In figure 4.23 and 4.24 there are two examples of person recognized by the neural network; figure 4.25 shows the analysis of the frames.

YoloV3 is set to recognize many objects so we decided to limit the network to person only; there is a special variable in the code that regulate the object list that the neural network can recognize that is `custom_object`.

As far card recognition is concerned we had to create our own model. Creation of a custom model can be very difficult but once again ImageAI help us in this thorny task. First of all we decide to use transfer-learning and so re-use a part of YoloV3 weights. This is a well-known practice in Deep Learning that can drastically reduce training times.

First of all we had to build the dataset containing cards. Our deck is composed of eight cards (figure 4.26 in the page 75) that are:

- Tree trash can: paper, plastic, glass.
- One umbrella card.
- Yes and no cards.
- Stop card.
- Repeat cards.

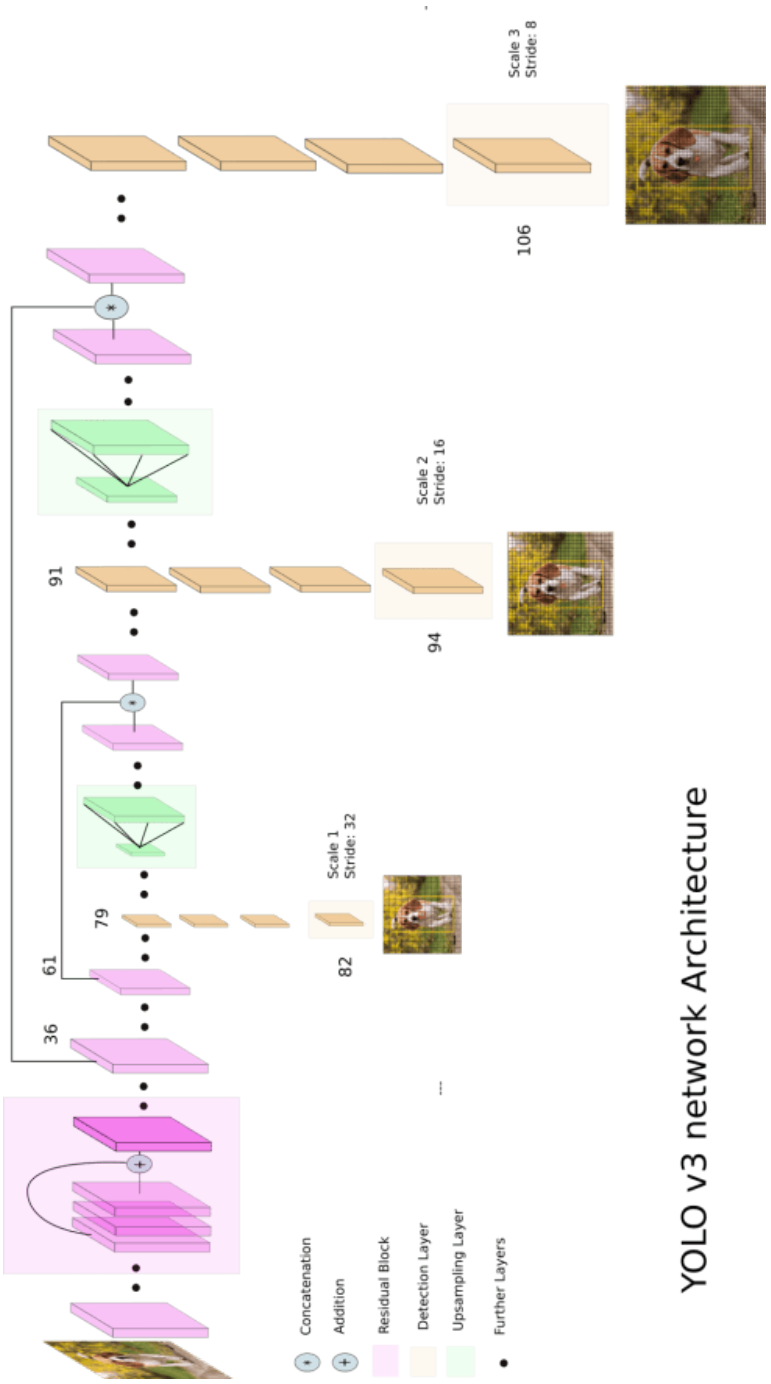


Figure 4.22: This figure shows YoloV3 architecture [57].

Figure 4.23: Output of one frame of the NN that recognize person.

Figure 4.24: Output of one frame of the NN that recognize person.

```
FOR FRAME  1
person  with  99.69850778579712 probability in position [180, 202, 543, 509]
------------END OF A FRAME --------------
Processing Frame :  2
FOR FRAME  2
person  with  99.54538941383362 probability in position [186, 203, 537, 509]
------------END OF A FRAME --------------
Processing Frame :  3
FOR FRAME  3
person  with  99.65084791183472 probability in position [186, 206, 539, 509]
```

Figure 4.25: Frame analysis of the NN for person.

Figure 4.26: In this figure there are shown the deck of cards that we used for the creation of the dataset and for the training of the Neural Network.

We gathered more then 200 images for object distributed in single and combined photos. This is the minimum recommended by ImageAI team.
After that we had to label each photo and this took a while since we had to do that by hand.
One the whole dataset was ready we had to face a big question: how do we divide dataset between training and validation?

Here experience came in help and we opted for k-fold cross validation with k=10.

Since the fact that training face is very demeaning in them of memory, GPU and time we decided to the well-known python service of Google: **Colab**.
Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.
Thanks to the already pretrained YoloV3 architecture we were able to write very few lines of code for creating and training the model; indeed we didn't have to compute all the layers of the network by hand. Our neural network was trained for more than 24 hours and we managed to do just 20 of the 100 epoch programmed for the training. We decided to stop the training at the 20^{th} epoch since the performance was already good for our purposes.
Now we'll talk about performances of our neural network. Even if we

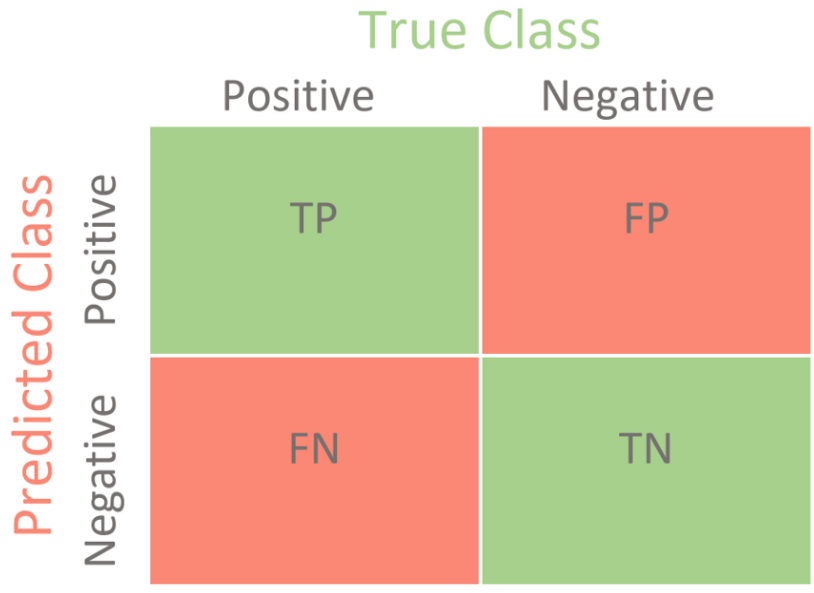


Figure 4.27: Confusion Matrix in the case of binary classification.

stop the training way before the proposed end, the performance of the network are more then good. We have an accuracy of 90% which is perfect. But let's talk about precision.
First of all we have to introduce the concept of confusion matrix [58]. A confusion matrix (figure 4.27) is a tabular way of visualizing the performance of your prediction model. Each entry in a confusion matrix denotes the number of predictions made by the model where it classified the classes correctly or incorrectly.

- True Positive: It refers to the number of predictions where the classifier correctly predicts the positive class as positive.
- True Negative: It refers to the number of predictions where the classifier correctly predicts the negative class as negative.
- False Positive: It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive.
- False Negative: It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative.

In figure 4.28 there is an example of multi-class confusion matrix. Now we can introduce the definition of precision 4.1.

$$Precision = \frac{TruePositive}{TruePositive + FalseNegative} \tag{4.1}$$

Even if our neural network has a very high accuracy, our precision is quite low. Despite the fact that our precision was low the network is still able to work because the probability assigned to the True Positive class is always bigger then the other classes. Some examples in figure 4.29 and 4.30. The second figure 4.30 is done with a low probability threshold (0.2) and so we can notice that the network start to say that each card can be anyone. Anyway the highest probability is always associated to the true object.
We can now discuss about the steps that we did to integrate ImageAI in HARMONI ecosystem. We create two different services: one for card recognition and one for person recognition. Talking more in general we create one service for YoloV3 model and one service for custom models.

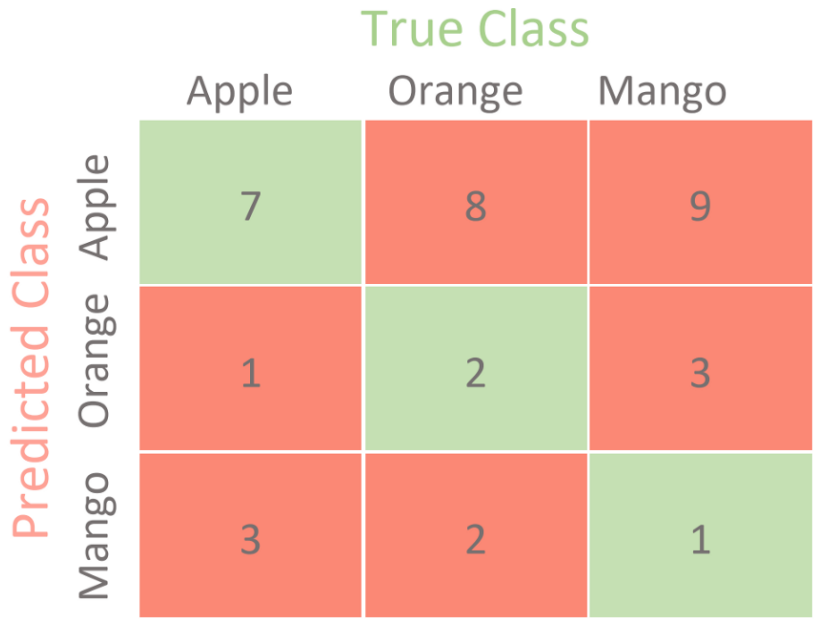


Figure 4.28: Example of confusion matrix in multi-class environment.

This because we want to give HARMONI users the possibility to create their own model and then use it. Indeed the parameters in the custom services are not fixed and developers can change them according to their needs.
Both services work as follow:

1. take one frame from camera service through the ros-topic
2. put that frame in input of the neural network and compute the result
3. repeat number 1. and 2. either a fixed number of time or till some object is detected

The reason we decided to reproduce a streaming recognition by using single frame detection is just one: **speed**.
Despite ImageAI allows developers to choose among several detection patterns: streaming recognition, frame recognition, different types of input format and so on; we noticed that streaming recognition had some problems concerning the time. While one frame was analysed

Figure 4.29: Output of one frame of the NN that recognize cards with a threshold of 0.95 .

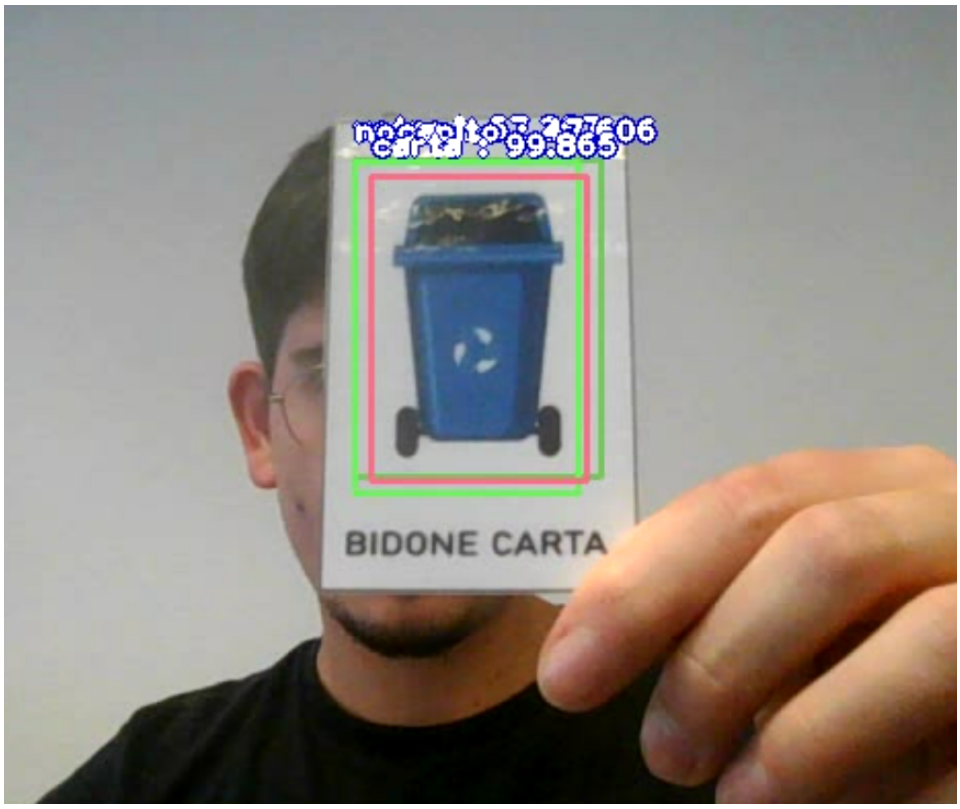


Figure 4.30: Output of one frame of the NN that recognize cards with a threshold of 0.2 .

```
Processing Frame :  11
FOR FRAME  11
carta  with  96.77308201789856 probability in position [262, 165, 391, 366]
------------END OF A FRAME --------------
Processing Frame :  12
FOR FRAME  12
carta  with  97.93567657470703 probability in position [261, 171, 389, 358]
------------END OF A FRAME --------------
Processing Frame :  13
FOR FRAME  13
carta  with  98.23046326637268 probability in position [264, 169, 390, 359]
------------END OF A FRAME --------------
Processing Frame :  14
FOR FRAME  14
carta  with  98.22984337806702 probability in position [263, 172, 390, 359]
------------END OF A FRAME --------------
Processing Frame :  15
FOR FRAME  15
carta  with  98.28354716300964 probability in position [262, 172, 390, 359]
------------END OF A FRAME --------------
Processing Frame :  16
FOR FRAME  16
carta  with  97.7271556854248 probability in position [263, 173, 389, 360]
------------END OF A FRAME --------------
Processing Frame :  17
FOR FRAME  17
carta  with  98.84545803070068 probability in position [268, 166, 393, 359]
------------END OF A FRAME --------------
```

Figure 4.31: Frame analysis of the NN that recognize cards with a threshold of 0.95 .

by the neural-network, the other ones where buffered in a queue and, since one frame take more than the frame rate to be processed the queue increase its size more and more. As a result within a few second the delay between the real frame and the frame analysed is too much to consider the hole system ”real-time”.
By using just one frame at a time we resolve the problem of buffer queue: we take a new frame form the camera only when the previous is correctly analyzed. In this way there is no delay, simply put we ”lose” some frames (the ones that arrives while a frame is analyzed by the neural network) but we keep the real-time assumption.
Figure 4.31 and 4.32 show the frames analysis of the figure 4.29 and 4.30 respectively. As said before if we reduce the probability threshold we can notice that the network outputs all the classes, despite this our algorithm is set to always take the object with the highest probability associated.
Each services have its own configuration parameters, we report here the parameters that can be changed by the users.

```
------------END OF A FRAME --------------
Processing Frame :  7
FOR FRAME  7
carta  with  99.93789196014404 probability in position [320, 156, 424, 306]
vetro  with  70.39631605148315 probability in position [320, 156, 424, 306]
nocapito  with  51.926714181900024 probability in position [318, 162, 427, 318]
plastica  with  29.222503304481506 probability in position [320, 156, 424, 306]
------------END OF A FRAME --------------
Processing Frame :  8
FOR FRAME  8
carta  with  99.77992177009583 probability in position [295, 182, 403, 326]
nocapito  with  48.3894944190979 probability in position [291, 175, 393, 333]
vetro  with  46.30752205848694 probability in position [295, 182, 403, 326]
plastica  with  22.517189383506775 probability in position [300, 194, 400, 325]
stop  with  20.923779904842377 probability in position [291, 175, 393, 333]
------------END OF A FRAME --------------
Processing Frame :  9
FOR FRAME  9
carta  with  99.89323616027832 probability in position [310, 145, 419, 286]
vetro  with  55.71275353431702 probability in position [313, 141, 428, 289]
nocapito  with  40.29734432697296 probability in position [309, 148, 419, 295]
plastica  with  34.29866135120392 probability in position [310, 145, 419, 286]
stop  with  24.3585005402565 probability in position [310, 145, 419, 286]
------------END OF A FRAME --------------
Processing Frame :  10
FOR FRAME  10
carta  with  99.90835785865784 probability in position [299, 165, 409, 318]
vetro  with  56.437575817108154 probability in position [297, 160, 410, 313]
nocapito  with  44.423866271972656 probability in position [299, 165, 409, 318]
------------END OF A FRAME --------------
```

Figure 4.32: Frame analysis of the NN that recognize cards with a threshold of 0.2 .

Plain yolo:

- **`frame_per_second`**: this controls the number of frame per second that the network will consider
- **`output_file_name`**: this is the name of the directory in which the output video will be saved
- **`log_progress`**: this is a boolean value and if it is set to `True` create a log
- **`minimum_percentage_probability`**: this is the minimum percentage for recognizing objects (threshold probability)
- **`return_detected_frame`**: this is a boolean value and if it is set to `True` allows to obtain the detected video frame as a numpy array

Custom yolo:

- **`frame_per_second`**: this controls the number of frame per second that the network will consider

- **output_file_name**: this is the name of the directory in which the output video will be saved
- **model_name**: this indicate the name of the model that developers will obtain at the end of the training phase. In our case this is `yolo_custom_card.h5`
- **json_name**: this indicate the name of the json file that developers will obtain at the end of the training phase. In our case this is `detection_config_card.json`
- **frame_detection_interval**: this control the object detection updated time in term of frames
- **log_progress**: this is a boolean value and if it is set to `True` create a log
- **minimum_percentage_probability**: this is the minimum percentage for recognizing objects (threshold probability)
- **return_detected_frame**: this is a boolean value and if it is set to `True` allows to obtain the detected video frame as a numpy array

In Appendix B you can find the documentation of ImageAI that we wrote for all HARMONI 's user.

Behaviours in PyTree

Of course in order to use ImageAI in PyTree we had to create its own leaf. We created two leaves, one for custom service and one for plain YoloV3. These two leaves work in the same way, the only difference is the server in which they perform the request.
The servers (services) handle the frames and the return values, by doing so the behaviours just have to perform the request and then wait for the answer.
For custom yolo leaf the answer of the service can be either `null` or `card-probability`. For plain yolo leaf the answer of the service can be either `null` or a list of `person-probability`.

These results are published on two different blackboard that are attached to the tow leaves.
For custom yolo we have:

- `card_detect/result` in write mode.

For plain yolo we have:

- `face_detect/result` in write mode.

These two leaves belong to the family of REQUEST ActionType. It means that when they are ready to perform the main task, that is ask for objects recognition of the frame, they will perform a REQUEST to the HARMONI service.

4.5.2 Amazon Lex

An important aspect that we had to take care was the *dialogue flow*. For dialogue flow we intend all possible branches that a speech between robot and user can go. In order to implement that we decided to use the well known service of amazon: **amazon lex** (this leaf is also called *bot*).
Amazon Lex [59] is an AWS service for building conversational interfaces into applications using voice and text. It uses the same deep learning engine that powers Amazon Alexa, enabling developers to build sophisticated, natural language chatbots that can be imported in applications. Amazon Lex provides the deep functionality and flexibility of natural language understanding and automatic speech recognition to enable you to build highly engaging user experiences with lifelike, conversational interactions and create new categories of products. This allows user to concentrate only on the semantic and on the main concepts of the dialogue without take care of the speech (text) analysis.
Now there are two different version of amazon lex: V1 and V2. We worked with version one.
In order to use amazon lex developers must familiarize with the following core concepts and terminology:

- **Bot** – A bot performs automated tasks such as ordering a pizza, booking a hotel, ordering flowers, and so on. An Amazon Lex bot is powered by Automatic Speech Recognition (ASR) and Natural Language Understanding (NLU) capabilities. Each bot must have a unique name within your account. Amazon Lex bots can understand user input provided with text or speech and converse in natural language.

- **Intent** – An intent represents an action that the user wants to perform. You create a bot to support one or more related intents. For example, you might create a bot that orders pizza and drinks. For each intent, you provide the following required information:

  - **Intent name** – A descriptive name for the intent. For example, OrderPizza. Intent names must be unique within your account.

  - **Sample utterances** – How a user might convey the intent. For example, a user might say ”Can I order a pizza please” or ”I want to order a pizza”.

  - **How to fulfill the intent** – How you want to fulfill the intent after the user provides the necessary information (for example, place order with a local pizza shop). We recommend that you create a Lambda function to fulfill the intent.

- **Slot** – An intent can require zero or more slots or parameters. You add slots as part of the intent configuration. At runtime, Amazon Lex prompts the user for specific slot values. The user must provide values for all required slots before Amazon Lex can fulfill the intent.
  For example, the OrderPizza intent requires slots such as pizza size, crust type, and number of pizzas. In the intent configuration, you add these slots. For each slot, you provide slot type and a prompt for Amazon Lex to send to the client to elicit data from the user. A user can reply with a slot value that includes additional words, such as ”large pizza please” or ”let’s stick with small.” Amazon Lex can still understand the intended slot value.

- **Slot type** – Each slot has a type. You can create your custom slot types or use built-in slot types. Each slot type must have a unique name within your account. For example, you might create and use the following slot types for the OrderPizza intent:
  - **Size** – With enumeration values Small, Medium, and Large.
  - **Crust** – With enumeration values Thick and Thin.

  Amazon Lex also provides built-in slot types. For example, AMAZON.NUMBER is a built-in slot type that you can use for the number of pizzas ordered.

Said that we create our own bot, intents and slots. In figure 4.33 there are two bots that we used during test and use phases.

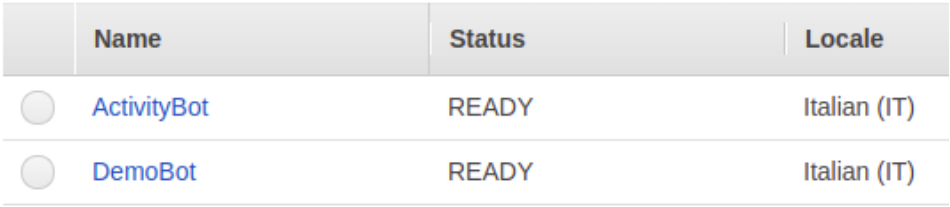


Figure 4.33: This figure shows our custom bots console of Amazon web services (AWS). The first bot called *ActivityBot* is the one used during the observations while the other one was used during the first focus group.

The first one, *ActivityBot* is the main bot where the two activities are saved. The other one *DemoBot* was used during focus group and testing phase.

In figure 4.34 there are all the intents belonging to the first bot. Each intent has ad interface that is like the one showed in figure 4.35, this intent is called *Plastica*.
Readers can notice that the first piece of information that has to be put in a intent is **Sample utterance**. This tag is very important and is related to the sentence that has to be said (or wrote) in order to trigger that intent.

In the *Plastica intent* you could also noticed that in the voice *Slots*

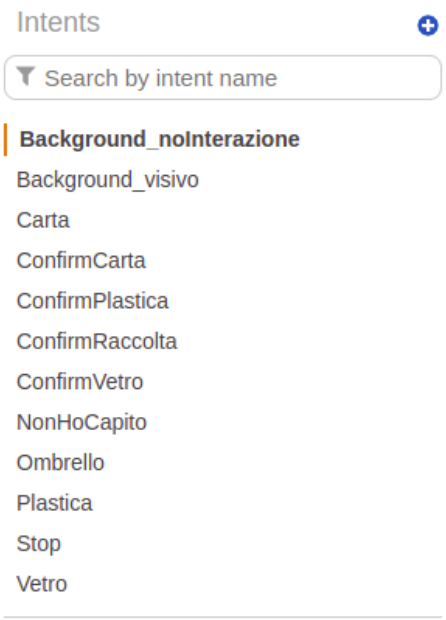


Figure 4.34: In this figure there are all the intents of ActivityBot bot.

there was a custom slot called *OggettoPlastica*. Figure 4.36 shows that slot in details. Readers can observe that the voice *Slot Resolution* (second voice of the figure) is set *Restrict to Slot values and Synonyms*, this means that *only* the values listed are considered "correct" values for the fulfillment of the slot. This because since there are some limitation in amazon lex we were forced to put exact values for the three materials (plastic, paper, and glass) otherwise answers of users were all equals because words like "can", "bin" or "thrash" can, reasonably, precede the material.
Confirmation prompt in figure 4.35 is used as an additional step before the fulfilment of the intent. In particular right before consider the intent fulfilled, the bot will ask for a *confirmation* question and here the user can either answer with a "yes" or a "no".
Response in figure 4.35 contains all the sentences that bot can say when the intent is fulfilled.

Last but not least, amazon lex have a special section for error handling (figure 4.37). *Clarification prompts* is the sentence that the bot will return when the state is *ElicitIntent* i.e. when bot can't understand

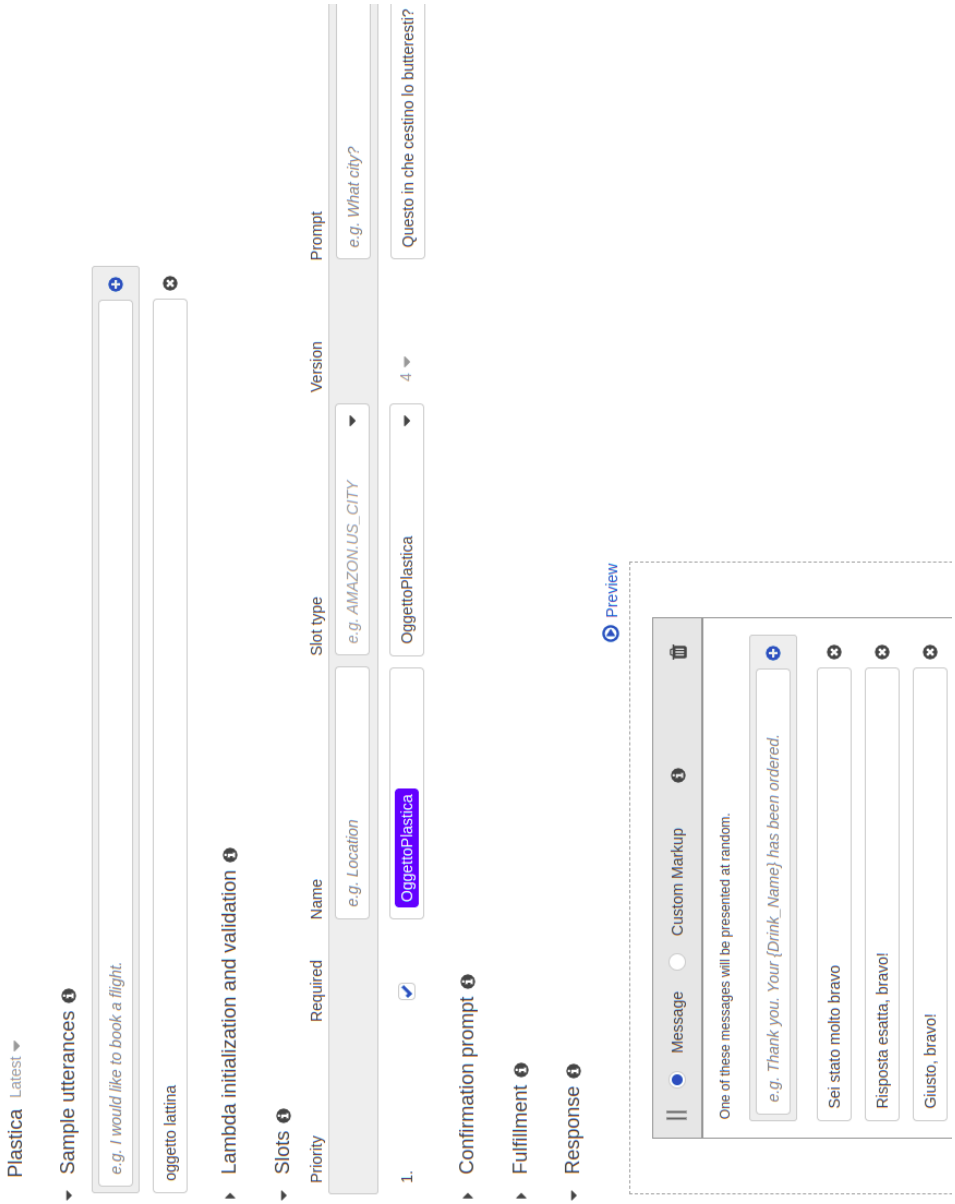


Figure 4.35: *Plastica* intent in details, users can modify parameters of intent here. These are all the intent can be triggered by the tree. As you can see the first piece of information that developers must add is the sample utterance that is the trigger sentence (or word) that has to be said in order to call that intent. All the other items are used to add slots, senteces and actions to the intent.

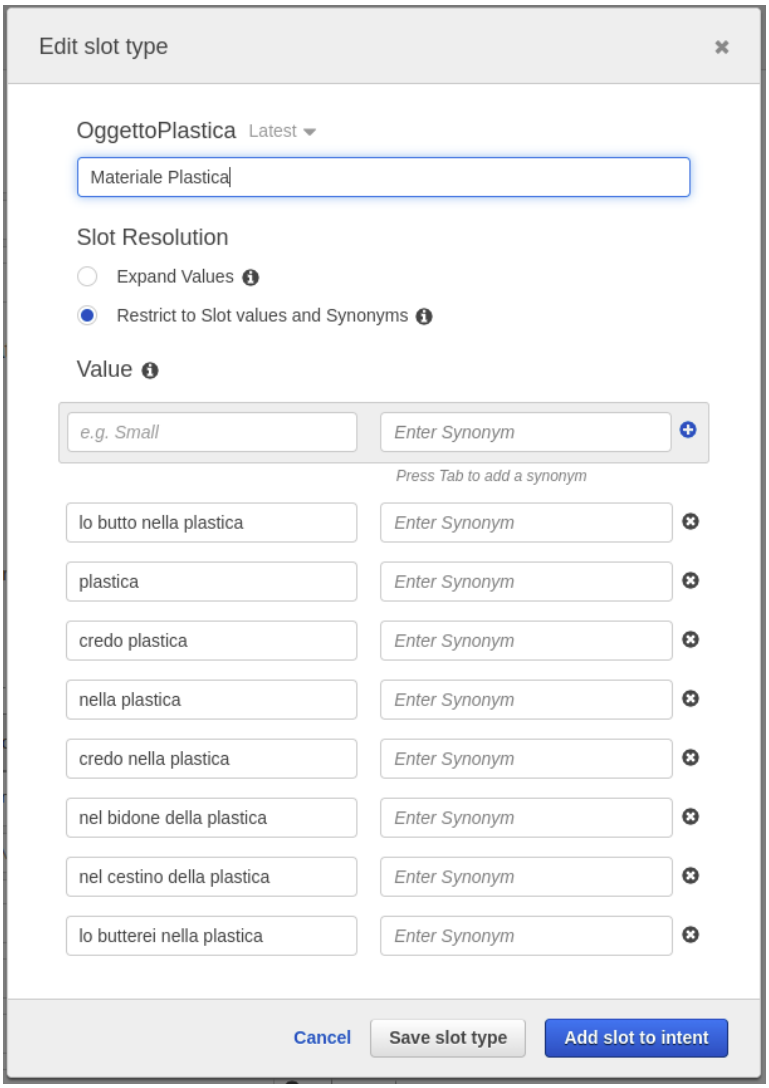


Figure 4.36: *OggettoPlastica* slot.

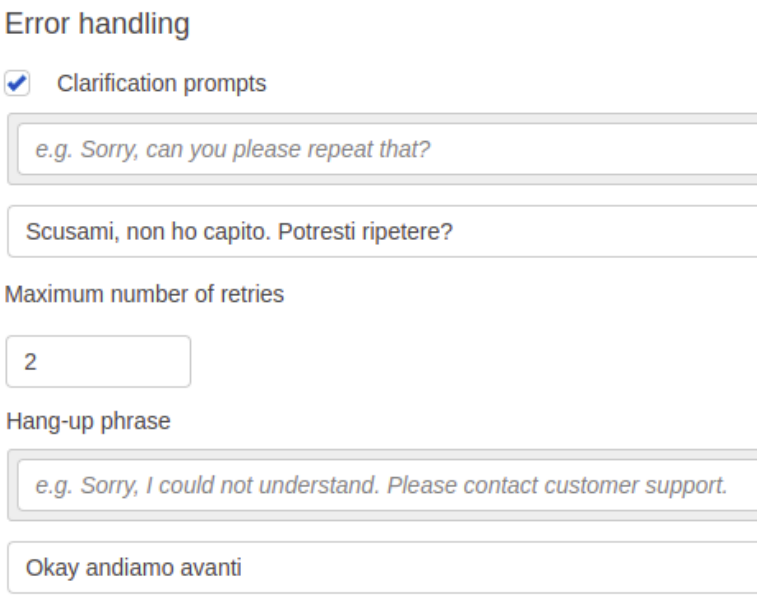


Figure 4.37: Our error handling section.

which is the current intent.
Maximum number of retries is a number that tells how many times the bot will try to understand the intent before return the state *Failed*; when this state is return *Hang-up phrase* will be also return as sentence for the robot.
The way in which developers can controls intents is through *state*. Indeed our leaf manage the states of the intent so as to understand what is happening to the conversation between user and robot and take appropriate actions.
An intent can be in one of several state that we want to report here:

- **Confirmed** - The user has responded ”Yes” to the confirmation prompt, confirming that the intent is complete and that it is ready to be fulfilled.
- **Denied** - The user has responded ”No” to the confirmation prompt.
- **None** - The user has never been prompted for confirmation; or, the user was prompted but did not confirm or deny the prompt.
- **ConfirmIntent** - The next action is asking the user if the intent

is complete and ready to be fulfilled. This is a yes/no question such as "Place the order?"

- **Close** - Indicates that the there will not be a response from the user. For example, the statement "Your order has been placed" does not require a response.
- **ElicitIntent** - The next action is to determine the intent that the user wants to fulfill.
- **ElicitSlot** - The next action is to elicit a slot value from the user.
- **Failed** - The Lambda function associated with the intent failed to fulfill the intent.
- **Fulfilled** - The intent has fulfilled by the Lambda function associated with the intent.
- **ReadyForFulfillment** - All of the information necessary for the intent is present and the intent ready to be fulfilled by the client application.

For the sake of order we decided to manage these state in a different PyTree leaf (Scene Manager Main) that we will explain later in this chapter.
We have two different leaves to control amazon lex service; the first one is called *trigger* and the other one is called *analyzer*. This distinction is because we have to perform request to amazon service more then one time in a single "run" of the tree.
The first one is used to trigger the intent that we want, this also means that the user will never trigger an intent; our activity is meant to have answer from users and not make request to the robot. So this leaf don't take in to account what the user does.
The second is used to analyze the speech of the user and then understand in which state the intent is. Simply put, this leaf take the speech of the user and use it as input of the conversational agent, as a response it will obtain an answer coming form the bot.
Trigger leaf has the following blackboards:

- `scene/utterance` in read mode.
- `bot/scene/utterance` in read mode.

Regarding analyzer leaf the blackboards are:

- `stt/result` in read mode.
- `card_detect/result` in read mode.
- `buttons/result` in read mode.
- `bot/analyzer/result` in write mode.
- `mainactivity/counter_no_answer` in write mode.

The main difference in this two leaves is that while the trigger leaf just perform a request with a "fixed" utterance, the analyzer leaf has also to decide where to take the answer of the user. We remember that an user can answer by using voice, buttons and cards (or he/she can not answer at all!).
The selection of the source is easily done by just looking at the blackboards values and by creating a priority order of the sources. The priority order that we decided was: (1) cards, (2) voice, (3) buttons.
Regarding values, we made services (plain yolo, custom yolo, speech-to-text and buttons) work as follow: if some results are found print them, otherwise print the string "*null*".
In this way the job of analyzer leaf is very easy: following the order that we previously decided, if some source printed something different form "*null*" use it, otherwise no answer is arrived.
Also these two leaves belong to the family of REQUEST ActionType, so they will perform a REQUEST to the HARMONI service.
It's important to emphasize that our amazon lex leaves control only the requests and the responses to and from HARMONI service. Then will be HARMONI service that will perform requests to the amazon service.
We can now show an example of dialogue between robot and user:

1. Amazon lex leaf will send to the server an utterance to trigger an intent, let's say that the utterance is "*oggetto plastica*" because it wants to trigger the intent "*Plastica*".

2. The bot will answer with the question "*Dove butteresti questo?*".
3. Suppose that at this point the child answer in a wrong way and say "*Nel vetro*".
4. Amazon lex leaf will send another utterance to trigger the the intent "*ConfirmPlastica*"
5. The bot will ask for the quest "*Sei sicuro? Non credi debba andare nella plastica?*" and since this intent is a confirmation intent the answer must be yes or no.
6. Suppose that the answer of the child is "*Yes*"
7. Amazon lex at this point will return *Confirmed* as status of the intent and it will say one of the sentences in the *Response* section, suppose: "*Esatto è la risposta esatta*!".

4.5.3 Amazon Polly Text-To-Speech

Amazon Lex is not the only amazon service that we used during our work. Indeed in order to implement text to speech we used Amazon Polly.
Amazon Polly is a service that turns text into lifelike speech, allowing users to create applications that talk, and build entirely new categories of speech-enabled products. Polly's Text-to-Speech (TTS) service uses advanced deep learning technologies to synthesize natural sounding human speech. With dozens of lifelike voices across a broad set of languages, you can build speech-enabled applications that work in many different countries.
In addition to Standard TTS voices, Amazon Polly offers Neural Text-to-Speech (NTTS) voices that deliver advanced improvements in speech quality through a new machine learning approach.
Amazon Polly server was already implemented in HARMONI so, in this case, our work was to create the leaf for PyTree.
Polly allows to generate speech from either plain text or from documents marked up with Speech Synthesis Markup Language (SSML). Using SSML-enhanced text gives you additional control over how Amazon Polly generates speech from the text you provide.

We used some of this tag so as obtain a better voice for the robot, we report here the two tags that we used most:

- *break* - this tag is used to add a pause.
- *emphasis* - this tag is used to emphasizing words.

Tags are used in the text sentence that is used as input of the speech-to-text service. For example the sentence:

"I already told you I <emphasis level="strong">really like </emphasis>that person."

Will be said by the robot emphasizing the two words *really like*.
Our implementation in PyTree of that service works as follow: it takes the sentence that the robot has to say and it perform a request to transform it in an audio. The request is taken by the server of HARMONI and when it has finished, the leaf will receive ad audio file in string form and it will print on the blackboard the value.
This leaf has the following blackboards:

- `tts/result` in write mode.
- `bot/trigger/result` in read mode.

This leaf belong to the family of REQUEST ActionType.

4.5.4 Google Speech-To-Text

So as to manage conversation between user and conversational agent we had to have a service that was able to transform speech in to text. The chose was on Google Speech Could service [60] .
A prototype of this module was already implemented in HARMONI but it barely worked so we had to fix some bugs before create the PyTree leaf.
After days of work we discovered that some problems that affected the HARMONI service was first of all problems concerning the version of service *google_cloud_speech*. Once upgraded the version of the google service and followed some hits on the documentation of google we made

that service worked.
Google Speech-To-Text (STT) works in two different ways: with audio data and with infinite streaming. We decided to use the streaming source since we thought that was better in term of activity and real time responses.
We want to report here one particular issue concerning the *chunk size* that caused some troubles during the debugging phase: chunk size is the smallest piece of information in which the audio streaming is divided; this size have to coincide in some sense with size in which the microphone convey the audio data or else the translation form speech to text will never work. Again we discover after some time that the chunk size has to be 1/10 the rate in which the microphone send audio data. The chaining of this simple number made the service works way better then before.
One remarkable adjustment that we did to the HARMONI server was the following: we added an extra parameter in the configuration file that is *max_duration*; this variable is an integer that represent second and controls the maximum time in which an user can **start speaking** before considering the voice answer *"null"*. This turned to be of great help when we had to decide of the communication channel that the user chose.
After that we could proceed in the creation of the leaf; this leaf belong to the family of REQUEST ActionType, so it's important to underline that when the leaf is ready to make the request it ask for google to transcribing of the speech while the microphone is open and it's recording.
The most attentive readers could understand that while this service make requests over time, and so it is some times active and some times inactive, the microphone service is always active and always record speeches.
This leaf has the following blackboard:

- `stt/result` in write mode.

4.5.5 Buttons

As we already said the third communication channel are buttons, buttons was the most atypical service since the fact that there is no server for this leaf. We decide to connect buttons directly to the robot through cable.
In order to create buttons we needed a chip so our research for the best component started; at the end we decide to use *Wemos d1 mini* form the family of *esp8266* (figure 4.38 and 4.39) [61].

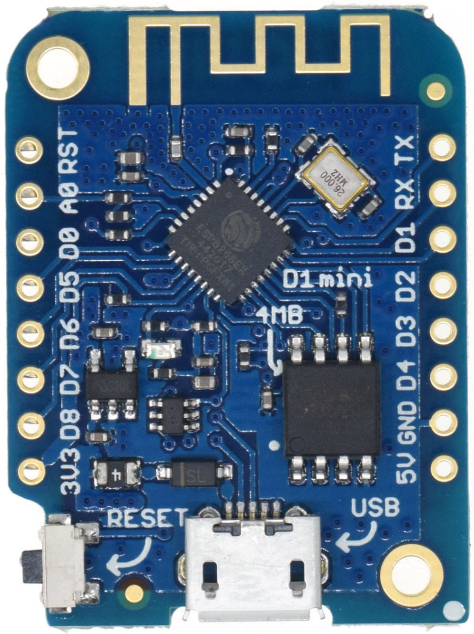


Figure 4.38: Wemos d1 mini front. This chip-set was used for the creation of the buttons

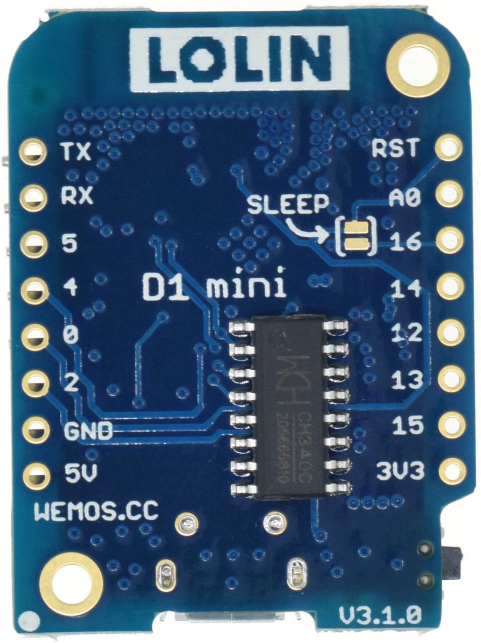


Figure 4.39: Wemos d1 mini back.

After welded buttons and wire on the board we used *Arduino* IDE to program the chip. Our program was very straightforward: when button A was pressed the chip print on the serial port the value *"button1pressed"* while when button B was pressed the message *"button2pressed"* was printed. In figure 4.40 you can see how buttons look like while in 4.41 and 4.42 you can see wires, relays and the wemos chip-set.
After that we could proceed in the creation of the leaf. It doesn't belong to any ActionType since there is no server request. The leaf, when activated, start to listen to the port where the chip is attached and then, if some answer arrives within *max_duration* seconds then it reports that message to its blackboard, otherwise the value *"null"* is printed.

This leaf has the following blackboard:

- `buttons/result` in write mode.

Figure 4.40: Black-box of buttons

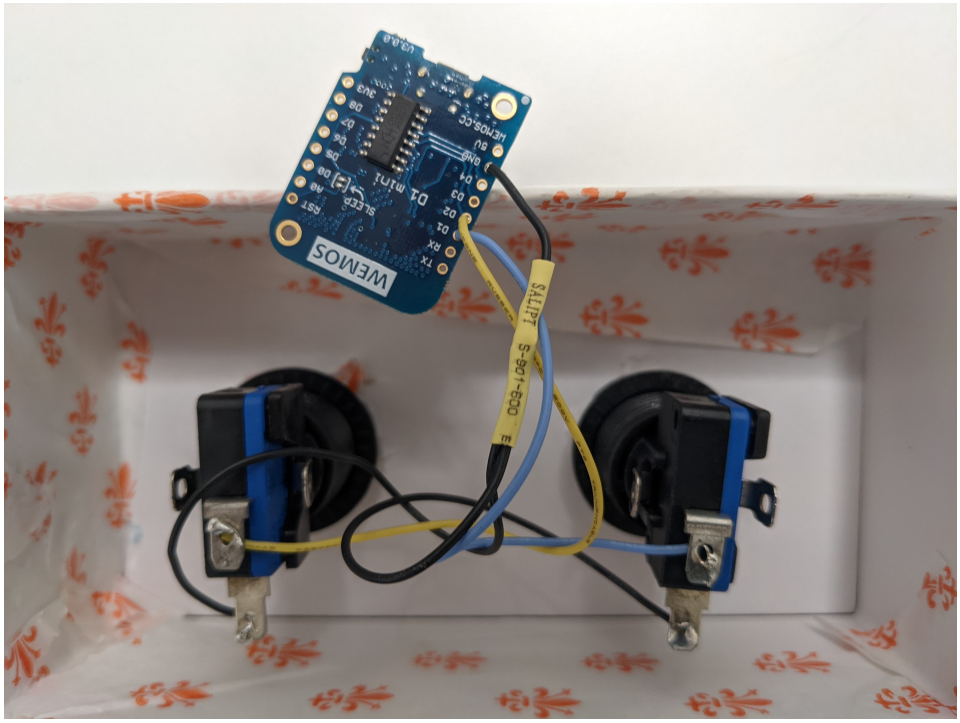


Figure 4.41: This is a view of buttons from the inside

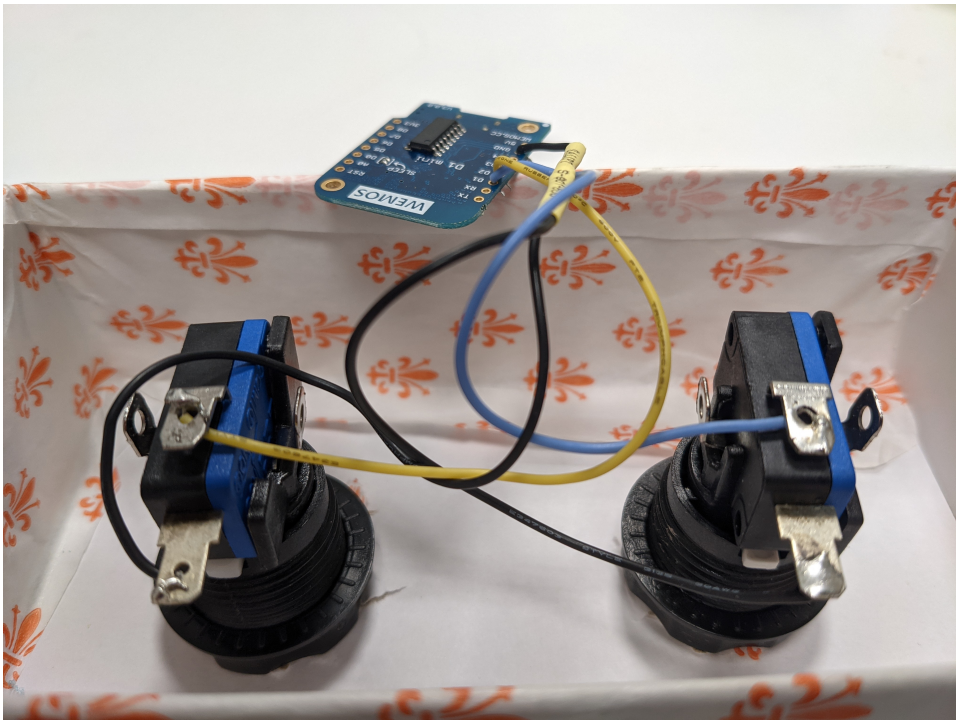


Figure 4.42: This is another view of buttons from the inside

4.5.6 Scene Manager

Scene Manager includes a family of leaves that we used to manage all the other leaves. There is one scene manager for each macro-section of the tree, more in details, there is one scene manager for the main activity one for the visual background and one for the interaction background.
We can say that these three leaves are the most important once since they set the initialize values of the blackboards for all the others leaves. There are no services call in these leaves and so they are non-blocking behaviours since, they can always decide what to do when they are ticked.
We decided to dived the activity in scene and so in order to keep track of the current scene and also all the pieces of information related to it we decide to create several json files. More in details we have three json files, one for each scene manager leaf, and the structure of the json is the same for all the three files. Actually regarding the scene manager of the main activity, we create more then one json because you can

map one story in one json. The more story you have the more json you need to create. Structure of a scene is presented in figure 4.43.

In addition to scene each json has also the so called *error_handling*


```
"scene" :   [
                {
                    "utterance"     : "E' mattina, ci siamo appena svegliati. Andiamo a fare colazione insieme.",
                    "face"          : "null",
                    "gesture"       : "null",
                    "image"         : "[{'component_id':'img_only', 'set_content':'https://firebasestorage.googleapis.com
                    "sound"         : "null",
                    "do_trigger"    : "False"
                },
```


Figure 4.43: An example of a scene.

scene: special scene that can be used where unexpected situation comes up. An example in figure 4.44.

We can notice that each section contains information for six leaves


```
"error_handling"    :
                    {
                        "riprendiamo_interacion"   :   {
                                                    "utterance"     : "Okay, riprendiamo da dove avevamo interrotto",
                                                    "face"          : "null",
                                                    "gesture"       : "null",
                                                    "image"         : "null",
                                                    "sound"         : "null",
                                                    "do_trigger"    : "False"
                                                },
                        "riprendiamo_visual"   :   {
                                                    "utterance"     : "Ah eccoti! Riprendiamo da dove avevamo interrotto",
                                                    "face"          : "null",
                                                    "gesture"       : "null",
                                                    "image"         : "null",
                                                    "sound"         : "null",
                                                    "do_trigger"    : "False"
                                                },
                        "risposta_corretta" :   {
                                                    "utterance"     : "Risposta esatta, ma non usarmi",
                                                    "face"          : "null",
                                                    "gesture"       : "null",
                                                    "image"         : "null",
                                                    "sound"         : "null",
                                                    "do_trigger"    : "False"
                                                },
```


Figure 4.44: Some examples of error handling.

that are:

- **utterance**: this is related to the sentence that the robot has to say **or** the utterance for trigger a specific intent.
- **face**: here goes a face expression for QT.
- **gesture**: here there is a gesture for QT

- **image**: here we can find an URL for a single image that the web page has to show
- **sound**: here goes a sound that the external speaker has to reproduce.
- **do_trigger**: this is a boolean value that tells if the string in the *utterance* is meat to be a trigger word for one intent or just a phrase that has to be said.

It is important to say that the value "null" is a special value that is used from the EitherOr idioms all over the tree to decide whether execute the specific behaviour or not.
The following subsection clarify what each single leaf does. Keep in mind that these three leaves, both with the design and the implementation of the tree, represent the biggest work for the realization of the recycling activity.

Main

This is for sure the biggest leaves we have, it controls all the blackboard of the leaves that are present in the subtree called *mainactivity*. The first thing that this leaves does (in the *setup* method) is to load the json file in a special variable, then its job is to control the flow of the story.
Every time this leaf is ticked it controls where we are in story and decide what to do: continue whit the story, repeat the last scene, handling some error that has occurred and so on.
This leaf has the following blackboard:

- `scene/mainactivity/scene_counter` in write mode.
- `scene/mainactivity/max_num_scene` in write mode.
- `scene/mainactivity/do_kid` in write mode.
- `scene/mainactivity/do_trigger` in write mode.
- `scene/utterance` in write mode.

- `scene/face_exp` in write mode.
- `scene/gesture` in write mode.
- `scene/image` in write mode.
- `scene/sound` in write mode.
- `bot/analyzer/result` in write mode.
- `bot/trigger/result` in write mode.
- `mainactivity/counter_no_answer` in write mode.
- `mainactivity/call_therapist` in write mode.
- `visual/inside` in write mode.
- `interaction/inside` in write mode.

Visual

The visual scene manager does the same job of the main scene manager but for visual background. Also this leaf, in the *setup* method, loads the json and then control the flow of the unexpected event where no one is in front of the robot. This leaf has the following blackboard:

- `scene/visual/scene_counter` in write mode.
- `scene/visual/do_trigger` in write mode.
- `scene/utterance` in write mode.
- `scene/face_exp` in write mode.
- `visual/inside` in write mode.
- `visual/finished` in write mode.
- `visual/call_therapist` in write mode.
- `bot/analyzer/result` in write mode.
- `bot/trigger/result` in write mode.

- `mainactivity/counter_no_answer` in write mode.
- `interaction/finished` in write mode.
- `interaction/inside` in write mode.

Interaction

The interaction scene manager does the same job of the visual scene manager but for interaction background. Also this leaf, in the *setup* method, loads the json and then control the flow of the unexpected event where the child doesn't interact with the robot. This leaf has the following blackboard:

- `scene/interaction/scene_counter` in write mode.
- `scene/interaction/max_num_scene` in write mode.
- `scene/interaction/do_kid` in write mode.
- `scene/interaction/do_trigger` in write mode.
- `scene/utterance` in write mode.
- `scene/face_exp` in write mode.
- `interaction/inside` in write mode.
- `interaction/finished` in write mode.
- `interaction/call_therapist` in write mode.
- `bot/analyzer/result` in write mode.
- `bot/trigger/result` in write mode.

4.5.7 Subtree Result

Sub Tree Result is a family of leaves like the Scene Manager family. Indeed, they are some similar aspects: there is one leaf for each subtree (main-activity, visual background, interaction background) and there are non blocking leaves since the fact that they don't perform any

request.
We can say that these three leaves are branches of scene managers, they perform some checks that scene manages will use but that they cannot compute by themselves. The main task of this modules is check if some final conditions are met and, in that case, prepare the tree to manage the end of that activity.
Simply put these leaves checks these two condition:

1. *Is the current scene the last scene?*
2. *Do I have to call the therapist?*

The following subsection clarify the differences.

Main

This leaf has the following blackboards:

- `scene/mainactivity/scene_counter` in read mode.
- `scene/mainactivity/max_num_scene` in read mode.
- `mainactivity/finished` in write mode.
- `mainactivity/call_therapist` in read mode.
- `scene/therapist_needed` in write mode.

Visual

This leaf has the following blackboards:

- `scene/visual/scene_counter` in read mode.
- `visual/finished` in write mode.
- `visual/call_therapist` in read mode.
- `face_detect/result` in read mode.
- `scene/therapist_needed` in write mode.

Interaction

This leaf has the following blackboards:

- `scene/interaction/scene_counter` in read mode.
- `scene/interaction/max_num_scene` in read mode.
- `mainactivity/counter_no_answer` in write mode.
- `interaction/finished` in write mode.
- `interaction/call_therapist` in read mode.
- `scene/therapist_needed` in write mode.

4.5.8 Web Page

This leaf is also known as *Projector* leaf; this because in the design phase we decided that the screen where show images was created by a projector.
This service was already implemented in HARMONI hence our work was only on PyTree leaf. This service works with HTML pages and CSS contents so what was showed on the projector was an HTML page with a single image on it. Each scene in the json file can define an image; the task of this leaf is to take the image and do a request to the HARMONI server and ask for printing that image.
What does this leaf is just a simple request, then the CSS engine will change the image accordingly to the request.
Since the fact that images are a lot we couldn't store all of them on the robot so we decided to use Firebase google service [62] as a server hence to upload images and then take them by the URLs.
This leaf has the following blackboard:

- `scene/image` in read mode.

4.5.9 Face

We have two different leaves to control face. The first one is used to make facial expression while the other one is assigned to control lips

synchronization.
Also in this case we had to create only the PyTree leaves since HARMONI made available for users all the services used to interact with face. Face is divided in three different services:

- mouth
- eyes
- nose

The first module, the facial expression one, use all of them while lips synchronization use just the mouth. Both of them belong to the family of REQUEST ActionType.
We will report in the following two subsection the blackboards that these two leaves use.

Facial Expression

This leaf has the following blackboard:

- `scene/face_exp` in read mode.

Lips Synchronization

This leaf has the following blackboard:

- `tts/result` in read mode.

4.5.10 Speaker and External Speaker

These two leaves control the voice of the robot and external sounds of the activity. They are very simple and both belong to the family of REQUEST ActionType.
In this case the server was already implemented in HARMONI so, one again we had only to create the PyTree leaves in order to use them in the tree.

Speaker

This leaf has the following blackboard:

- `tts/result` in read mode.

External Speaker

This leaf has the following blackboard:

- `scene/sound` in read mode.

4.5.11 Microphone

This leaf belongs to the family of DO ActionType, this means that the first time it is ticked a request for the opening of the service is done and then, in the following ticks, there are just controls that the service is still alive and working properly.
Microphone is always opened and it always publish pieces of data that it captures, it will be the Speech-To-Text service that will take data only when the request is performed by google service.
This leaf doesn't have any blackboard since no messages need to be sent over other leaves.

4.5.12 Camera

Also this leaf belongs to the family of DO ActionType, so its functioning is very similar to the Microphone.
It is always opened and it always publish pieces of data that it captures, it will be the plain yolo service, and custom yolo service, that will take data only when the request is performed.
This leaf doesn't have any blackboard since no messages need to be sent over other leaves.

Chapter 5

Exploratory Research

"Exploratory research is really like working in a fog. You don't know where you're going. You're just groping. Then people learn about it afterwards and think how straightforward it was." — Francis Crick

5.1 Method

Exploratory research is defined as a research used to investigate a problem which is not clearly defined. It is conducted to have a better understanding of the existing problem, but will not provide conclusive results. For such a research, a researcher starts with a general idea and uses this research as a medium to identify issues, that can be the focus for future research. Such a research is usually carried out when the problem is at a preliminary stage. It is often referred to as grounded theory approach or interpretive research as it used to answer questions like what, why and how: it forms the building blocks of an overall research project. The data gathered from these research can be qualitative or quantitative.
For the proposed solution and field of study, we have used most of all qualitative ones.
Qualitative research methods are used to gather non numerical data. It is used to find meanings,to observe a target audience behavior, opinions, or the underlying reasons from its subjects. These methods are unstructured or semi structured. The sample size for such a research

is usually small and it is a conversational type of method to provide more insight or in-depth information about the problem.

5.1.1 Study design

There are different exploratory research methods, we have settled for focus group and observation.

- focus group: is a research technique used to collect data through group interaction. The group comprises a small number of carefully selected people who discuss a given topic. Focus groups are used to identify and explore how people think and behave, and they throw light on why, what and how questions
- observation: is a method to observe and describe the behavior of a subject. As the name suggests, it is a way of collecting relevant information and data by observing. It is used in cases where you want to avoid an error that can be a result of bias during evaluation and interpretation processes. It is a way to obtain objective data by watching a participant experience with the proposed solution.

Focus group

The reason why we have chosen to do a focus group was because we had the possibility to do it with people related to our target, so with NDD but with fairly high cognitive level. We thought that who better than them could help us understand the pros and cons of what we were coming up with. It would have been an occasion both for us to have feedback from them but most of all to make them feel important, considered and active participants in something that would have helped other people like them.
For the focus group we have prepared a demo in order to show the three different interaction modes with the robot: verbal,cards,buttons. We've gave them also three pictures to better understand and also to choose the modality that they have preferred to interact with the robot (figure 5.1).

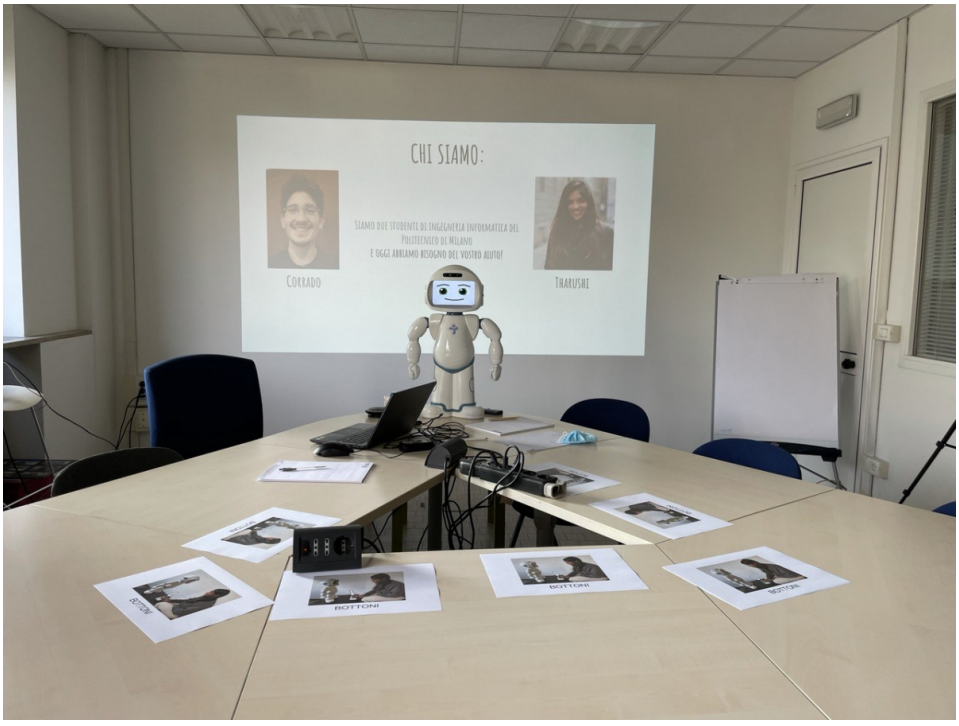


Figure 5.1: We have given everyone three pictures representing each interaction modality

We have prepared four different activities/games, including recycling, to be evaluated by the participants. We asked from them how much in a scale 1 to 5, they like the activity presented and which modality they prefer in that activity to interact with the robot. The activities proposed were related to concentration, memory and learning daily services:

- ”raccolta differenziata”: this is our recycling activity where the child has to answer robot questions saying where to throw the rubbish.

- ”memory dell’orto”: game in which you have in a first moment a grid 3x3 showing all the vegetables in it, then the child has to remember where every vegetable was in order to answer robot questions when the grid is empty (Figure 5.2).

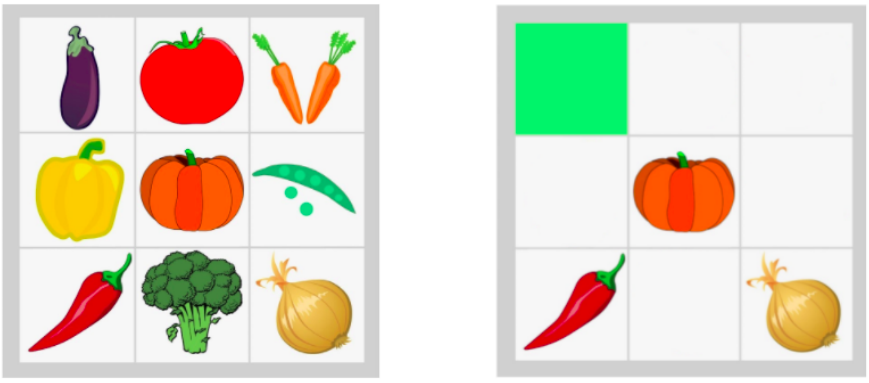


Figure 5.2: On the left there is the grid at the beginning of the game and on the right there is an example of the question asked

- ”segnali stradali”: here you have to learn and answer some question related to road signs (Figure 5.3).

Figure 5.3: The robot asks what is the meaning of this road sign

- ”contiamo insieme” : the robot shows the child some pictures and for each one asks how many elements there are in a group (Figure 5.4).

Figure 5.4: The robot asks how many boots are there in the first picture? 2,5 or 4?

Figure 5.5: First focus group with Fraternità e Amicizia centre at i3lab at Politecnico

Figure 5.6: Second focus group with CSE Collage centre

Observation

The observation phase was based on people with NDD, testing the activity that we have created. Our objective was to collect as much information as possible for each participant related to how they feel to interact with the robot, what type of interaction they prefer, how difficult is the activity, what are the typical behaviours, critical aspects etc.
Observation has been done in three meetings with distinct level of disorders, cognitive capacities. We went to two different centers "Fraternità e Amicizia" and "Collage" bringing with us all the necessary materials: robot, projector, pc, buttons, external speaker and cards (Section 5.1.4).
Since we were in front of disparate disorders and levels, we have extracted also a short story from the full one, where we remove the interlude and asks less questions.

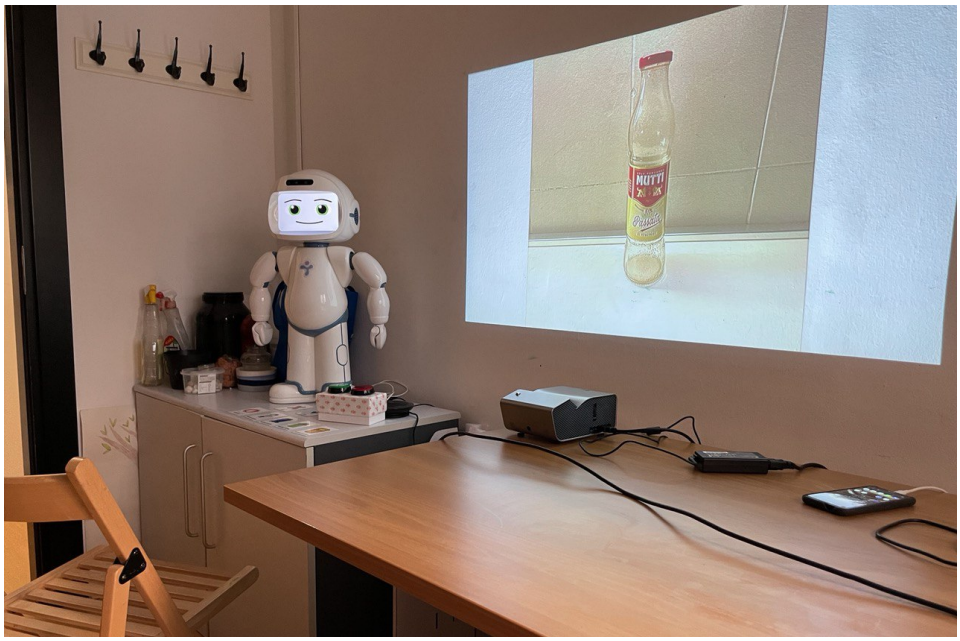


Figure 5.7: First day of observation, setting in Fraternità e Amicizia centre

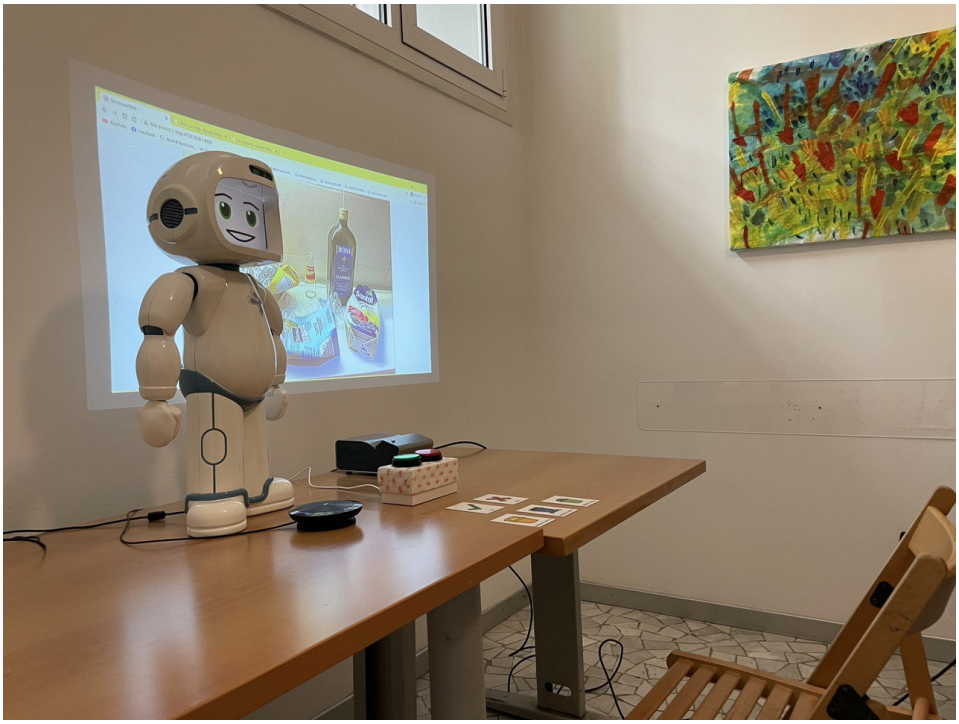


Figure 5.8: Second day of observation, setting in Fraternità e Amicizia centre

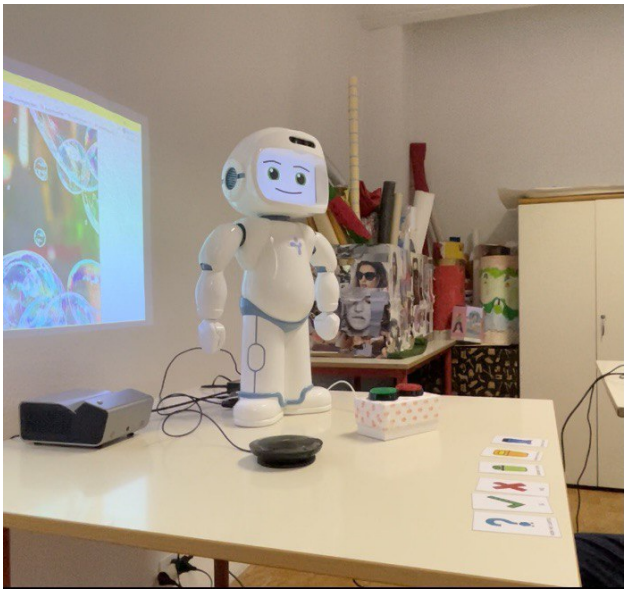
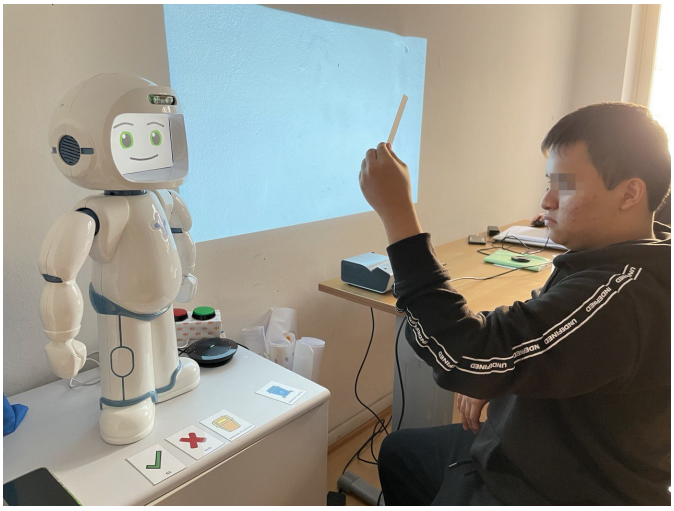


Figure 5.9: Third day of observation, setting in CSE Collage centre



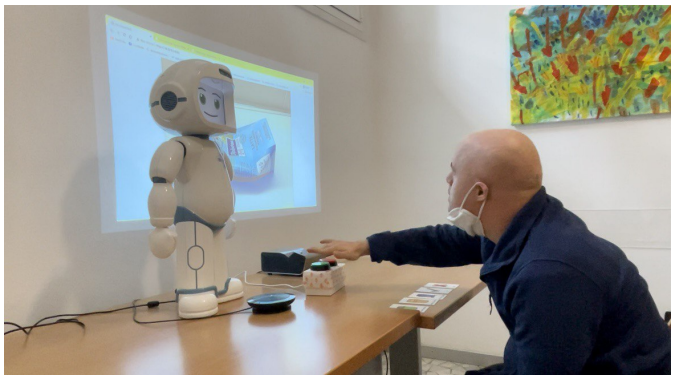


Figure 5.10: Two participants from Fraternità e Amicizia during the first and second day of observations

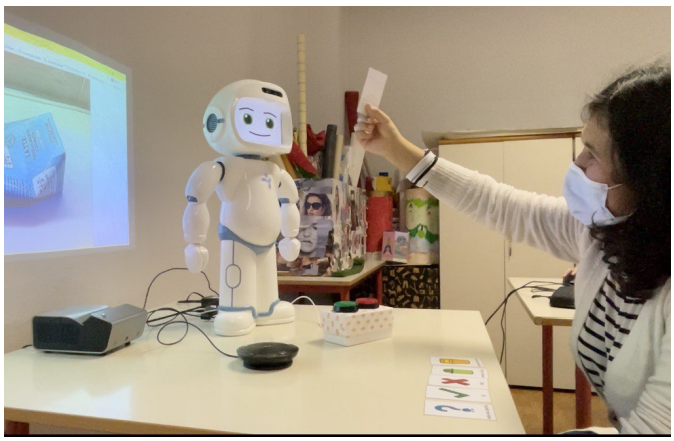


Figure 5.11: One participant of CSE Collage during the third day of observation

5.1.2 Questionnaires

In this section we report the questionnaires that we did during both focus groups and observations.
Questions done during the first focus group:

1. *Experience with robot*
2. *I would feel relaxed talking to robots*
3. *I would feel uncomfortable if I was given a job where I had to use robots*
4. *Are robots similar to humans?*
5. *Are robots scary?*
6. *Are robots weird?*
7. *Are robots ugly?*
8. *Are robots dangerous?*
9. *Are robots aggressive?*

Questions done during the second focus group:

1. *Experience with robot*
2. *I would feel uncomfortable if robots really reveal any emotions*
3. *I would feel relaxed talking to robots*
4. *I would feel uncomfortable if I was given a job where I had to use robots*
5. *If robots had emotions, I would be able to befriend them*
6. *I hate the idea that robots or artificial intelligence are making judgments about things*
7. *Are robots reliable?*
8. *Are robots qualified?*
9. *Are robots interactive?*
10. *Are robots similar to humans?*
11. *Are robots happy?*
12. *Are robots clumsy?*
13. *Are robots scary?*

14. *Are robots weird?*
15. *Are robots ugly?*
16. *Are robots dangerous?*
17. *Are robots aggressive?*

Questions done after the observations:

1. *Have you ever had experiences with augmentative communication with cards*
2. *How much experience do you have with devices to talk to for example google home alexa?*
3. *How much did you enjoy the experience with the robot?*
4. *Did you enjoyed cards?*

5.1.3 Participants

Focus Group

We have conducted two focus group:

- **First day - Fraternità e Amicizia**: first focus group conducted with six participants from Fraternità e Amicizia Social Cooperative Onlus of Milan. We had the opportunity to meet them in i3lab at Politecnico (Table 5.1).
- **Second day - CSE Collage**: second focus group conducted with ten participants from Social Educational Centre Collage of Milan. For this meeting we went to their centre (Table 5.2).

Person Id	Gender	Age	Diagnosis
P1	F	28	Psychomotor retardation, FG syndrome, associated with mild relational disorders and multiple dysphonologies, hypovisus
P2	M	25	Mental insufficiency, mild bipolar affective syndrome
P3	M	31	Psychotic OCD associated with mild mental retardation, celiac disease
P4	M	28	Atypical autism
P5	F	30	Mixed behavior and emotional disorder, moderate mental retardation, drug-resistant epilepsy
P6	M	27	Autistic syndrome

Table 5.1: Focus Group participants from Fraternità e Amicizia

Person Id	Gender	Age	Diagnosis
P7	F	34	Mild cognitive impairment
P8	F	28	Severe cognitive impairment Obesity, metatarsalgia from overuse
P9	F	26	Severe cognitive impairment due to trisomy 14
P10	M	41	Complex malformative syndrome, with cognitive impairment, severe asthma, and multiple allergies.
P11	F	37	Severe cognitive impairment
P12	M	35	Middle cognitive impairment
P13	M	38	Severe cognitive impairment
P14	M	42	Down syndrome
P15	F	44	Severe cognitive impairment
P16	F	31	Congenital encephalopathy associated with obesity, refractory epilepsy, moderate-severe cognitive impairment, and behavioural disorders

Table 5.2: Focus Group participants from CSE Collage

In table 5.3 there is reported the structure and the steps that we did in focus groups and the average time used to do them.

Item	Approx. duration
Welcome and introduction	5m
Discussion about interaction method	10m
Demos	15m
Questions about games	10m
Brainstorm about new games	10m
Survey on robots	10m

Table 5.3: Structure of the focus group discussion.

Observation

We have conducted three observation sessions: two in Fraternità e Amicizia in two different centres and one in Collage

- **First day - Fraternità e Amicizia**: first observation conducted with 6 participants at Fraternità e Amicizia Social Cooperative Onlus centre (Table 5.4).

- **Second day - Fraternità e Amicizia**: second observation conducted with 7 participants at Fraternità e Amicizia Social Cooperative Onlus centre (Table 5.5).

- **Third day - CSE Collage**: third observation conducted with 7 participants at CSE Collage centre (Table 5.6).

Person Id	Gender	Age	Diagnosis
P17	F	35	Epilepsy, severe cognitive impairment
P18	M	29	Epilepsy, cognitive impairment
P19	M	34	Down syndrome
P20	M	20	Generalized developmental disorder cognitive impairment
P21	F	33	Microcephaly, bilateral neurosensorial deafness. Growth and ponderal deficit
P22	F	21	Severe cognitive impairment Stereotypical movements

Table 5.4: First observation participants from Fraternità e Amicizia

Person Id	Gender	Age	Diagnosis
P23	M	41	aftermath of perinatal encephalopathy with moderate-severe deficit
P24	M	34	down syndrome, impeded vision, bilateral hearing deficit
P25	M	38	moderate cognitive impairment
P26	M	53	epilepsy severe cognitive impairment, dyslalia due to perinatal encephalopathy prenatal, encephalopathy with mild oligophrenia and cognitive impairment.
P27	M	52	Emotional and relational disorders
P28	F	35	cognitive impairment due to marfanoid habitus
P29	M	59	severe post-vaccination encephalopathy with severe cognitive impairment.

Table 5.5: Second Observation participants from Fraternità e Amicizia

Person Id	Gender	Age	Diagnosis
P7	F	34	Mild cognitive impairment
P8	F	28	Severe cognitive impairment Obesity, metatarsalgia from overuse
P9	F	26	Severe cognitive impairment due to trisomy 14
P10	M	41	Complex malformative syndrome, with cognitive impairment, severe asthma, and multiple allergies.
P11	F	37	Severe cognitive impairment
P12	M	35	Middle cognitive impairment
P13	M	38	Severe cognitive impairment

Table 5.6: Observation participants from CSE Collage

5.1.4 Tools

The materials used for the focus group are:

- power point presentation describing the main topics covered during the focus group. It also contained the different activities/games proposed (Section 5.1.1).
- three images representing each interaction modality (Figure 5.1).
- pallets that go from 1 to 5, used for voting the activities and questions (Figure 5.12).
- one questionary about the proposed activities and one related to HRI.
- QTrobot (Figure 3.1).
- buttons
- cards

- projector (Figure 5.13).
- speaker (Figure 5.14).
- pc

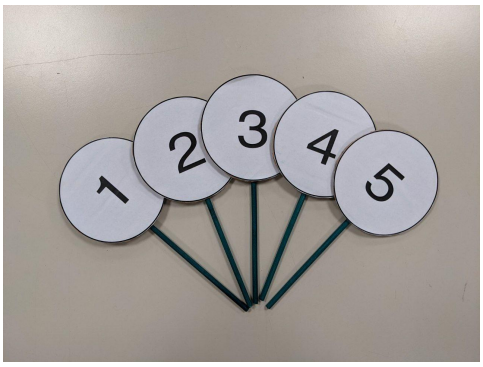


Figure 5.12: This picture shows pallets used during focus groups.

Figure 5.13: LG Led Portable Projector

Figure 5.14: Speaker Jabra 510 used with robot during observations

The materials used for the observation are:

- QTrobot (Figure 3.1).
- buttons
- cards
- projector (Figure 5.13).
- speaker (Figure 5.14).
- pc

5.1.5 Hypothesis

Considering that often children with NDD, especially ASD, have some pattern of language use and behaviours that don't let them to communicate in the most efficient way (section 3.6.1) such as repetitive or rigid language or uneven language development, **we have hypothesized that buttons and most of all, cards would have been the most used interaction paradigms**. This because once they have associated the card to its meaning, they just have to use it to answer; there is no need of communication audibly, use of correct words and grammar, intonation and so on. We also thought that cards have as their point in favor the fact that they are characterized by simple and colorful pictures that can attract their attention and therefore push the participant to use them in order to communicate.

5.1.6 Data analysis

In order to obtain meaningful results and do a simple and practical analysis a posteriori, we made up survey and questions in which answers are given in the *Likert Scale* from 1 to 5.
The survey is a composed by general questions on robots and it can be divided in two parts, the first one is on questions regarding behaviours between robot and person, the second one is on questions regarding just the robot. The question that we made in the two focus group are reported in (section 5.1.2) on page 119.

In order to make questionnaire more enjoyable from the point of view of the participants, we provided five vote-sticks per participants (Figure 5.12). We read questions aloud and then we transcribed on paper the answer of each participant.
Of course we also reported speeches of the participants both when they wanted to add something more on questions and when the questions expected open answers. Indeed a lot of tips and advice to improve activity come up.
As far observations are concerned, we thought that the creation of surveys wasn't the right choice since they are too constrained so we proceeded by reporting in natural language what happened. Indeed during observations some unexpected situation went out so prefixed surveys would have been difficult to fit.
During the activity of each participant we wrote on paper his/her behaviour and all the meaningful actions of the latter. We also recorded the entire activity of each participant (if and only if the participant accepted to be recorded through valid certification) and then we reanalyzed all the videos in order to add and adjust what we wrote previously.
We also prepared some fixed questions whose answer was in *yes or no* form for each participant. We reported pieces of information like whether he/she managed to reach the end of the activity or not, whether he/she needed help to end the activity or not.
After reported behaviours of users during the activity, we tried to extrapolate also the main themes which are shared features, ideas, perceptions, pros and cons during the individual activity session and put them in a graphical representation using a word cloud (Figure 5.24).
At the end of the observations we also ask for closed questions in Likert Scale style; questions are showed in section 5.1.2 on page 120.

5.2 Results

Focus Group

In figure 5.15 there are the results of the survey that we made in the first focus group. Regarding the second focus group, figures 5.16 , 5.17,

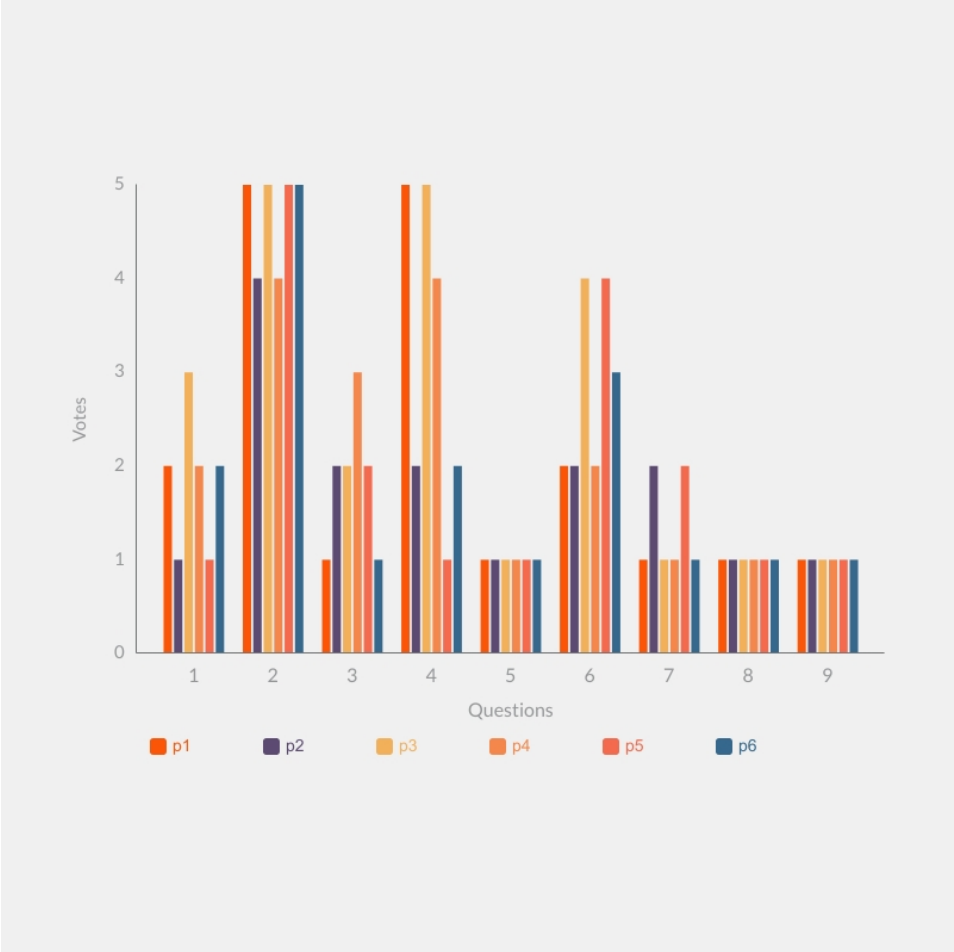


Figure 5.15: This plot represents the answers of the survey on the first focus group (section 5.1.2, first set of questions). On the x axis there are the questions, on the y axis there are the possible answers (from 1 to 5) and px are the participant of the focus.

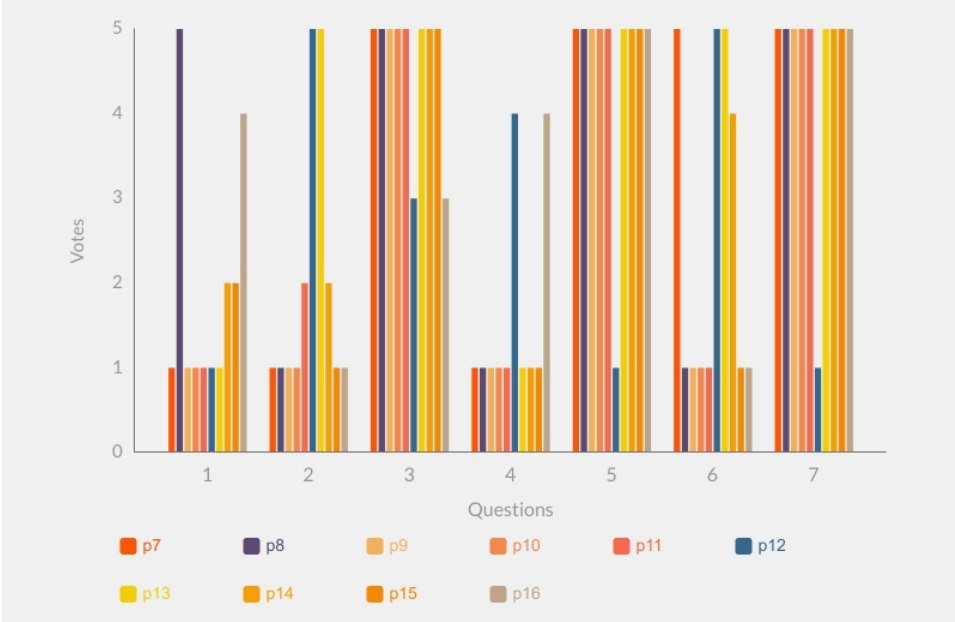


Figure 5.16: This plot represents the answers of the survey on the second focus group (section 5.1.2, second set of questions). On the x axis there are the questions, on the y axis there are the possible answers (from 1 to 5) and px are the participant of the focus.

5.18 show the results. We divided results in three figures since in these focus group questions were 17 instead of 9.

The following four figures 5.19, 5.20, 5.21, 5.22 show the results about the "questions about games" phase reported in table 5.3. The first pie chart in each figure shows votes regarding modalities preferred while the second one shows, in a range between 1 to 5, how much people enjoy the game. Results are reported in percentages.

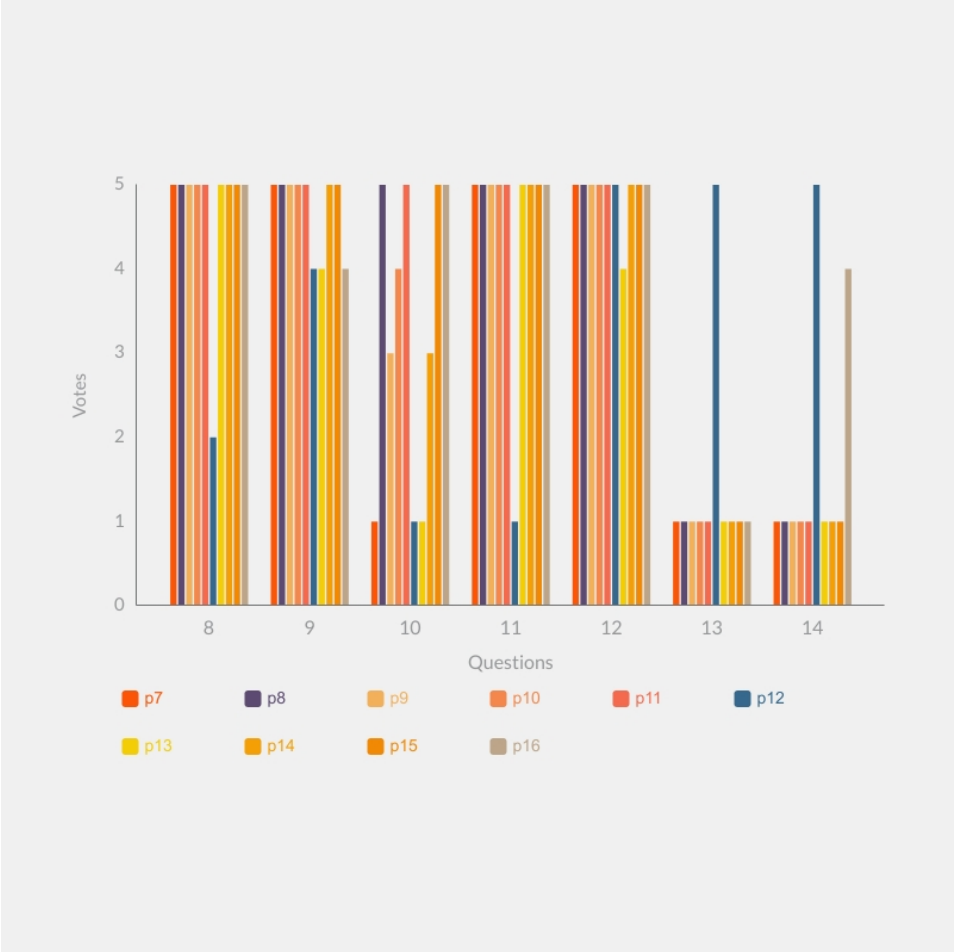


Figure 5.17: This plot represents the answers of the survey on the second focus group (section 5.1.2, second set of questions). On the x axis there are the questions, on the y axis there are the possible answers (from 1 to 5) and px are the participant of the focus.

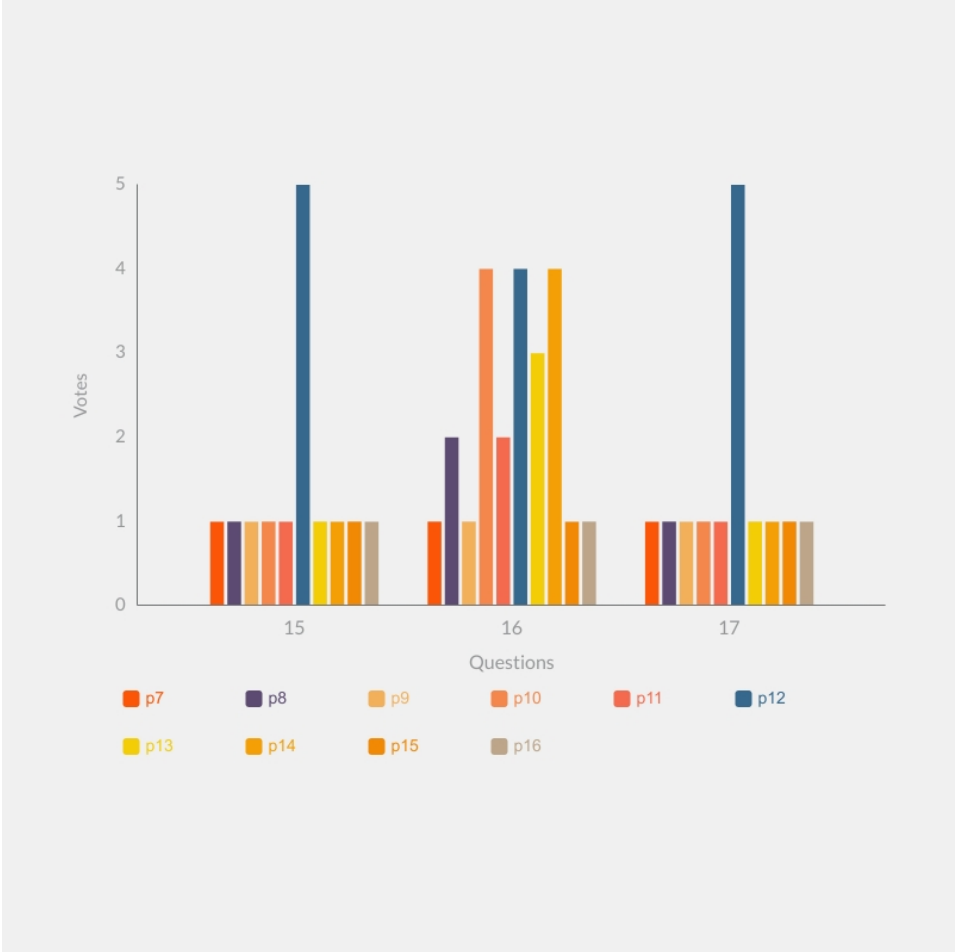


Figure 5.18: This plot represents the answers of the survey on the second focus group (section 5.1.2, second set of questions). On the x axis there are the questions, on the y axis there are the possible answers (from 1 to 5) and px are the participant of the focus.

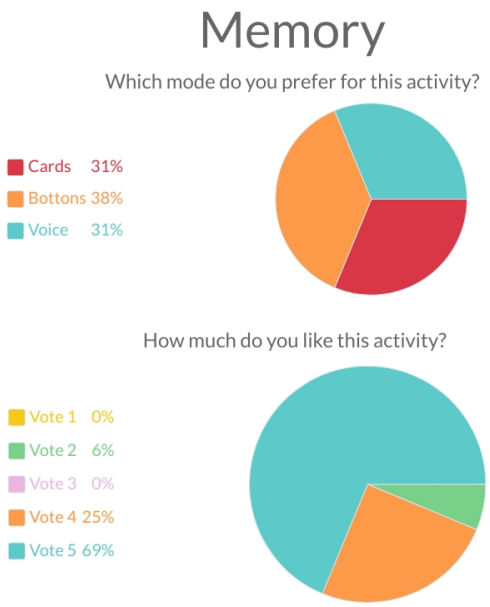


Figure 5.19: This two pie charts are referred to memory game, the first one reflect the preference about modalities while the second one reflect the preference about the game itself.

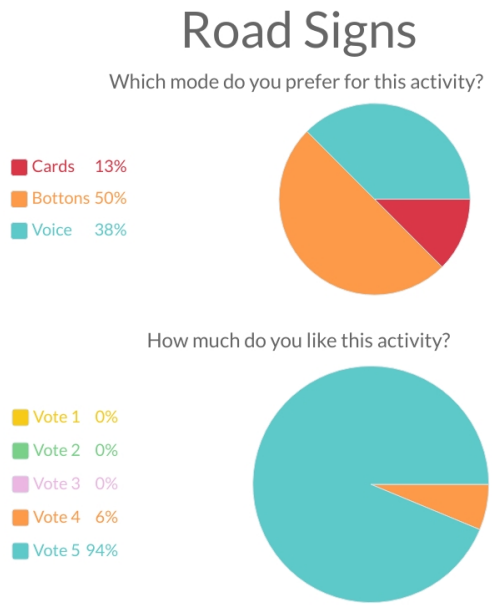


Figure 5.20: This two pie charts are referred to road signs game, the first one reflect the preference about modalities while the second one reflect the preference about the game itself.

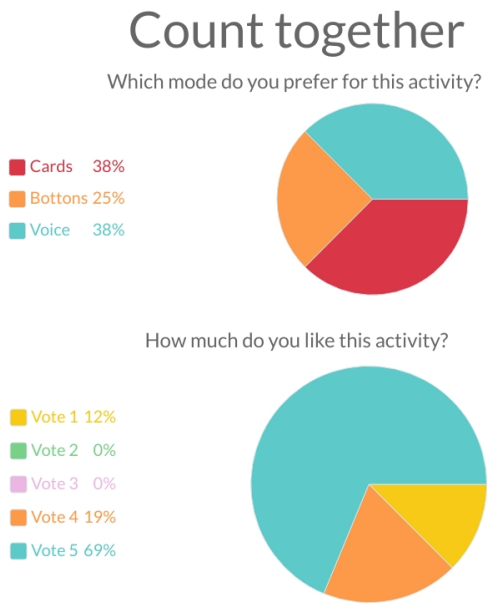


Figure 5.21: This two pie charts are referred to count together game, the first one reflect the preference about modalities while the second one reflect the preference about the game itself.

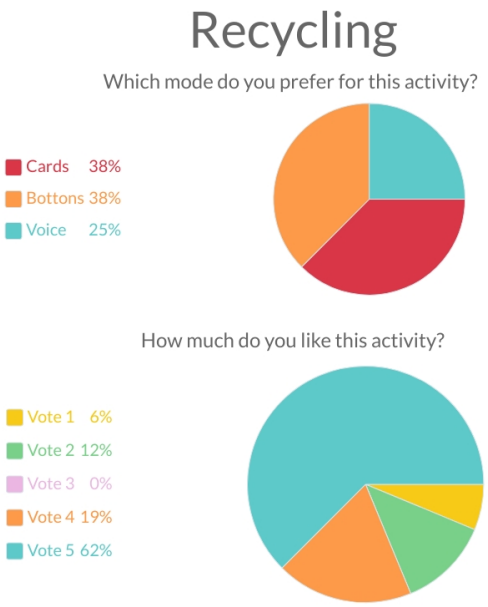


Figure 5.22: This two pie charts are referred to recycling game (the one that we implemented), the first one reflect the preference about modalities while the second one reflect the preference about the game itself.

Observation

In Table 5.7, Table 5.8, Table 5.9 there are reported all the participants that has taken part to the individual session activity:

- Modality preferred : which interaction paradigm was the preferred one among cards, voice and buttons for interacting with the robot.
- Activity completed: yes/no answer to say if the activity has been completed by the participant.
- Help Needed : yes/no answer to say if the participant was autonomous or needed help from the therapist or from us during the activity to interact and answer to the robot.

First day - 18.10.21

Person ID	Modality Preferred	Activity completed	Help Needed
P17	Cards	yes	yes
P18	Voice	yes	yes
P19	Cards	no	yes
P20	Cards	no	yes
P21	Cards	yes	yes
P22	-	no	yes

Table 5.7: First day of observation at Fraternità e Amicizia. The table shows for each participants the interaction paradigm preferred, if the activity has been completed and if the participant needed help.

Second day - 19.10.21

Person ID	Modality Preferred	Activity completed	Help Needed
P23	Cards	yes	yes
P24	Voice	yes	yes
P25	Buttons	no	yes
P26	Voice	yes	yes
P27	Cards	yes	yes
P28	Voice	no	yes
P29	Cards	yes	yes

Table 5.8: Second day of observation at Fraternità e Amicizia. The table shows for each participants the interaction paradigm preferred, if the activity has been completed and if the participant needed help.

Third day - 21.10.21

Person ID	Modality Preferred	Activity completed	Help Needed
P7	Cards	yes	no
P8	Cards	yes	no
P9	Cards	yes	no
P10	Cards	yes	no
P11	Cards	yes	no
P12	Voice and Cards	yes	no
P13	Voice and Cards	yes	no

Table 5.9: Third day of observation at CSE Collage. The table shows for each participants the interaction paradigm preferred, if the activity has been completed and if the participant needed help.

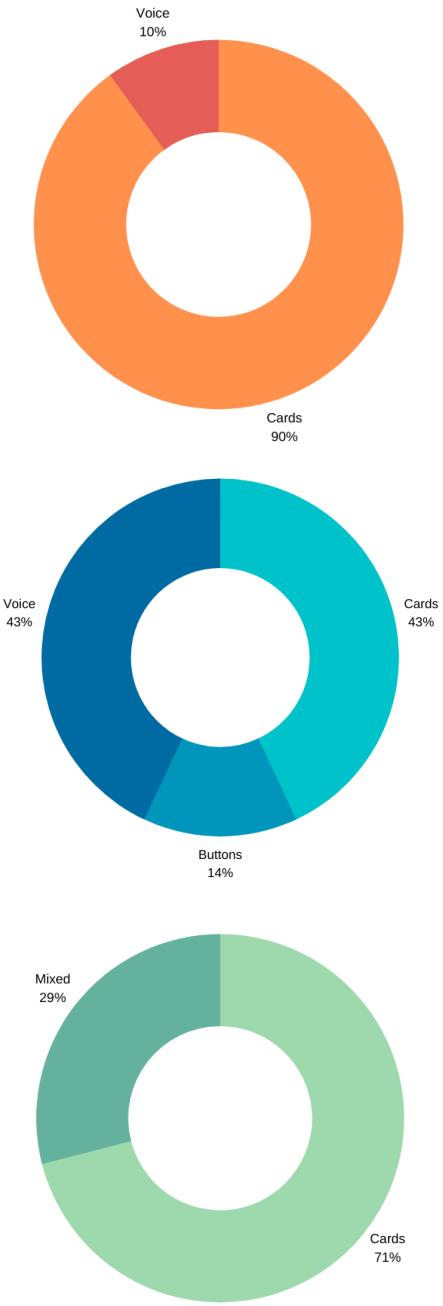


Figure 5.23: Doughnut charts representing the interaction modality preferred during the individual sessions. The first one is related to the first day, the second one to the second day and the third one to the third day. As it can be noticed there is also ”Mixed” it refers in particular to the used of both voice and cards

We decided to create a word cloud in order to detect and show the most important topics that came out during observations. Figure 5.24 shows the result of this process while Figure 5.25 and Figure 5.26 show results of the questions made at the end of the observations.

Figure 5.24: The word cloud is a graphical representation of word frequency. This one shows the result of our analysis of the themes come out during observations. Themes are drawn from individual activity of each participants with QTrobot. The dimension of each word is proportion to the number of time the same concept was repeated during all the observation sessions.

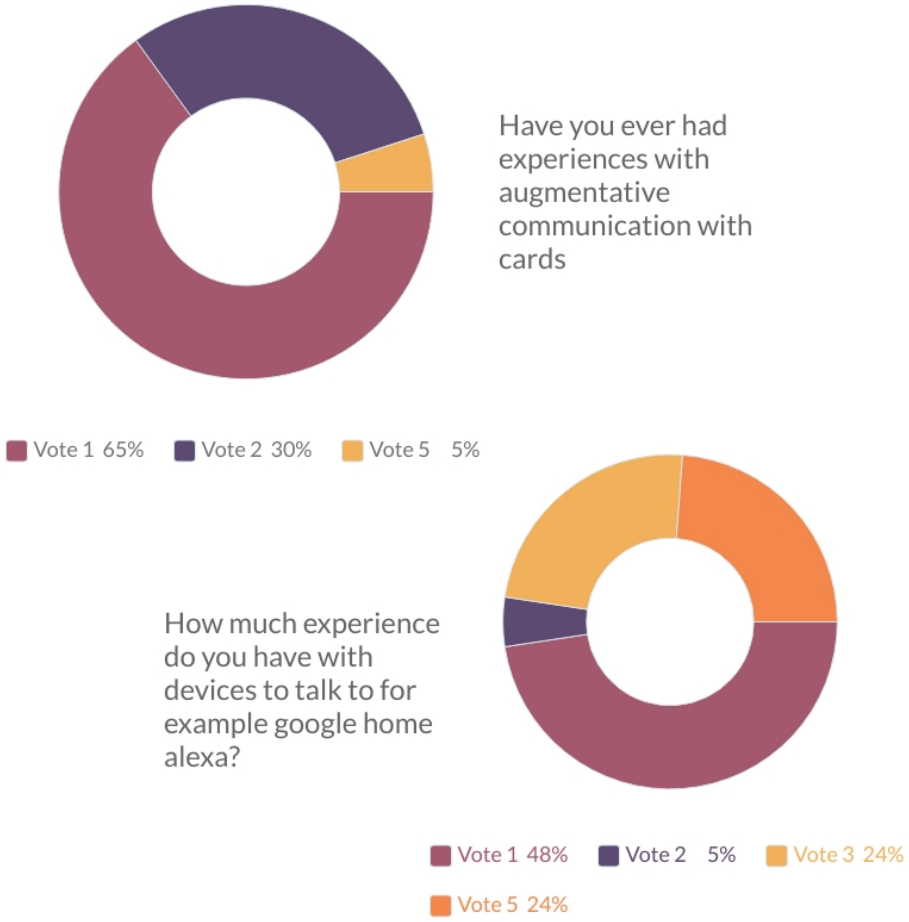


Figure 5.25: Here there are the results of the first and second questions made after observations (section 5.1.2 third set of questions).

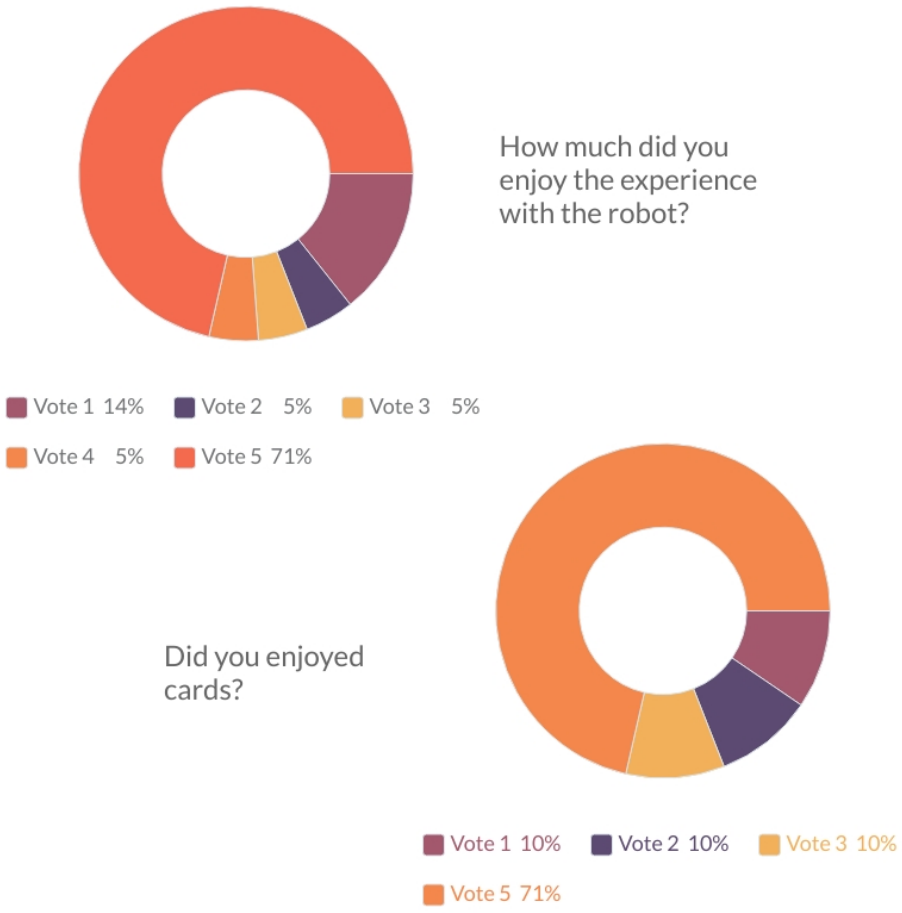


Figure 5.26: Here there are the results of the third and fourth questions made after observations (section 5.1.2 third set of questions).

5.3 Discussion

According to the results obtained from the surveys comes out that this segment of the population still fairly today cut off from technology since they aren't able to have a lot of experiences with robot, augmentative realities or communication devices (Figure 5.25). It is regrettable given that most of the participants enjoyed the interaction with the robot and would be comfortable to work or simply play with it (Figure 5.26). As had emerged from previous research, the participants have pleasure in interfacing with the robot, they feel less judged and interact with it as it is a friend. It can be said that the activity that we have implemented fits well to a target that has a basic knowledge about the social activity proposed, in this case, recycling whereas it would have been better to implement another activity, such as "memory dell'orto" in which the participant doesn't have to rely on his/her knowledge or has to understand different concepts for the first time. Furthermore is important to consider that each activity depends on the cognitive level of the participant in fact as it can be seen from Table 5.7 and Table 5.8 there are some participants that have been able to complete the activity and others that haven't. For sure every participants has its peculiarity and needs and that's why it would be better to create an activity that fits each of them.
Regarding the interaction paradigm that is preferred, certainly it mustn't go unnoticed that most of the participant have preferred to communicate through tools such as cards and buttons instead of voice. Also the participants with the highest cognitive level, have used cards to better express the answer. It may be the novelty, it may be the fact that it resembles a game, the fact that they like pictures and colors but we can say that the cards definitely helps them to interact with the robot and to express themselves. In this way we can say that with the observation session we have confirmed our hypothesis about what would have been the most used interaction paradigm.

5.3.1 Limitations

In this section we are going to report both activity and hardware limitations that we have encountered :

- **Interaction**: one of the biggest limitations that we found was caused by the fact that users are not used to interact with robot and use this kind of technologies. Since that we had to explain several times what participants were going to do (speak with a robot, show card to answer, etc) but some times after that phase users were still disoriented. It could have been useful to have a training phase in which the participant gets familiar with the robot, understand how to distinguish and show the cards and so on. This was not possible due to restricted time for the experimentation but for sure if this type of solution, that is, use of a SAR for doing such activities, is adopted in the centres, there would be certainly better results and participants can absorb and show their improvement and capabilities.

- **Customization**: each participants was different from the other and different were the cognitive levels so for some of them would have been better to have a more simple activity or a different approach. For example, some participants before saying where to throw the rubbish, tried to analyze the object and then told the robot what was the object projected on the wall. For the latter its better to have a recognition phase guided by the robot, for others there would be also the possibilities to add other bins and concepts.

- **Selection of one modality**: for some participants three different ways of interaction were too much and this can be a source of confusion. So the selection of one modality could simplify the activity.

- **External sound**: we didn't manage to add external sounds due to some limitations of the QTrobot's internal components.

- **Buttons**: we implemented buttons just to answer *yes or no* questions. We think that this could have been a reason why buttons

were not used a lot. Maybe buttons could have been used even more than cards by participant if we would have implemented a button for each recycling bin.

- **Amazon Lex**: when there is a yer-or-no question (*Confirmation-Intent*) lex *only* expects exactly either *yes* or *no*. There aren't other possibilities and this is a limitation since some user can also answer by using words like *surely*, *certainly*, *why not*, *I don't think so* etc etc.
  Other limitations of amazon lex were in the understanding of natural language that sometimes is not as good as it should be. Of course we are aware that the last problem is one of the today's challenges of Machine Learning algorithms.

- **Volume**: the maximum volume of the internal speaker sometimes was to low and participants didn't even hear questions.

- **Room**: specific room are required in order to do these kind of activity. A room should be noiseless and simple so as to put the participants at ease. If a projector is used, the room should be dark enough to let the screen be visible clearly. We also encountered some delay due to internet connection, services like amazon polly, lex and google need a good internet connection to work properly so the activity should we done in a rooms where the internet connection is good and stable.

Chapter 6

Conclusions and Future work

6.1 Conclusion

The thesis work shows very important results in the field of interactions between socially assistive robot and people with neurodevelopmental disorders. The main outcome is that traditional communication methods, such as verbal one, don't always fit the need of our target. Some of them are more encouraged to interact with a robot using cards or buttons.
Before doing any test we made an hypothesis: cards and buttons will be the interaction methods that participant will use more. The reason why we did this assumption is because not all people with NDD feel comfortable in speaking; it's also possible that some of them have language impediments and so voice won't be an option at all.
In addition cards and buttons can be easily paired to actions and objects. Once a participant get the meaning of a card, or a button, it is easy to him/her express that concept: just press the button or show the card related to that concept. Last, but not least, cards and buttons are also funny and enjoyable objects and this is something that is not negligible.
After the test phase we were glad to discover that our hypothesis were confirmed by results in fact the most used and appreciated interaction paradigm were cards.
Results showed that using a robot for conducting these kind of activities increase involvement and fun from users and tester. Feedback from

tester also showed that they liked a lot the cards, the shape, pictures and colors. These latter made the activity more similar to a game and so more enjoyable from the point of view of participants.
It's important to say that even if augmentative communication methods (like cards and buttons) allow a more concise and direct communication, it can be successfully used when there are some simple concepts to reinforce or when some concepts has to be repeated over and over again. These methodologies are not as good as in the previous cases when it comes to elaborate complex concepts or sentences, for the moment voice is still the best method we have for these latter.
Wrapping up results, we can state that *Voice* is a good communication method that works well in a lot of situation but it can't be used in any context and not every person feel comfortable in using it; *Buttons* and *Cards* seems to be promising alternatives for both user experience and information convey.
In the end we can state that using cards and buttons as communication channels between robots and people with NDD can improve both participation and interaction: people can express their willing in a simpler and clearer way. We hope that our contribution in this field helped to get closer people and technology and made communications easier and within everyone's reach.

6.2 Future work

Here we would like to talk about improvements and future works that can be done in order to empower our research and thesis work.
First of all we think that an enlargement of the number of people that would test our activity can be a great chance to confirm and support our results. We are aware of the lack of researches and studies in the field of multi-modal communication methods between people with NDD and robots, indeed our work was based on exploratory research and so it can be a starting point for others works that can lay their fundamentals on more structured concepts.
Testing with participants gave us a lot of interesting improving sparks and we'd like to talk about some of them.

First of all we think that for some participant having all the three communication methods usable is too much and it can bring feelings of disorientation and frustration. So a selection of the modalities at the start of the activity is a easy and good improvement that can be done in short term.
We think that the person detection module can be combined with a face detection one in order to make the detection more precise and reliable. Face detection module can also give hint on the emotional profile of the user. Indeed we think that an extra background module can be added so as to manage situations in which the user appear to be bored or distracted.
Also a recognition of the pose can be useful, understanding mental state of a person in not an easy job so the more modules we can add the more accurate will be the comprehension of the mental state. Pose will recognize the actual orientation of the body in the space and also joints of harm, legs and neck.
Gesture of the robot and external sounds could be added in the activity in order to make it more immersive and natural. The skeleton of these modules have already been arranged and added to the activity but for now they are disabled modules since there weren't the hardware counterpart. Once hardware will be added there is just a matter of adjust some parameters.

Appendix A

PyTree

PyTree is one among the several ways to implement behaviour trees.
A tree in pytree is made of two elements: nodes and leaves.
Nodes can be of different nature but in general they belong to the family of compositors (sequence, parallel, selector). The management of these elements it's not something that users have to do, everything has already been done by the creators of pytree.
Leaves come from a single element called "behaviour" and these are the components that users can create. Even if pytree made some basic behaviours available for us (you can find some of them here[1]), developers are required to create their own behaviours (skeleton of behaviour[2]).
We provide some leaves that represent the pytree version of clients of HARMONI.

A.1 Setup

In order to use PyTree import the python library by running the following command:

```
$ pip install py_trees
```

(optional) If you want to create and see dot_tree you must install pydot and graphviz.

[1] <https://py-trees.readthedocs.io/en/devel/modules.html#module-py_trees.behaviours>

[2] <https://py-trees.readthedocs.io/en/devel/behaviours.html#skeleton>

For pydot run:

```
$ pip install pydot
```

For graphicviz run:

```
$ sudo apt install graphviz
```

A.2 Usage

The execution of a tree is done by ticking it.
A tick starts from the root and then is propagated down in the tree up to leaves. Leaves will perform their tasks and in the end compute a state that will be returned and propagated back until the root.

As mentioned before a tree is composed of nodes and leaves. Since all the leaves are behaviours that have their own class we suggest creating a different script for the body that will import all the behaviours that it needs (an example here[3]).
You can notice that after creating the tree all you need to do is call the function `behaviour_tree.tick_tock()` that will take care of ticking the tree. Parameters `pre_tick_handler` and `post_tick_handler` are used to lick functions that will be executed respectively before and after the tickling of the tree. In general they are used to see the status of the tree and additional information like the content of the blackboards.
We suggest adding a special element of the family of visitors that is called *snapshot visitor*. This component is used combined with `display.unicode_tree` to also show the statue of the visited element in the tree. You can add it by adding the following lines in the declaration of the behaviour tree:

“‘

snapshot*visitor* = *py*trees.visitors.SnapshotVisitor()

behaviour*tree.visitors.append(snapshot*visitor) “‘

and then adding in the parameter `visited` in `dispaly.unicode_tree` like that

```
py_trees.display.unicode_tree( root=behaviour_tree.root,
visited=snapshot_visitor.visited)
```

[3] <https://py-trees.readthedocs.io/en/devel/trees.html#skeleton>

You can also decide to manually tick the tree by running `behaviour_tree.tick()` instead of `tick_tock()` function.

PyTree offers a lot of components and useful tools that developers can use in their trees but here we will mention **compositors** and **blackboards**. We recommend reading the documentation for understand all the concepts: documentation[4]

Compositors

Composites are responsible for directing the path traced through the tree on a given tick (execution). They are the factories (Sequences and Parallels) and decision makers (Selectors) of a behaviour tree.
Composite behaviours typically manage children and apply some logic to the way they execute and return a result, but generally don't do anything themselves. Perform the checks or actions you need to do in the non-composite behaviours. composites[5]

Blackboards

Blackboards are not a necessary component of behaviour tree implementations, but are nonetheless, a fairly common mechanism for sharing data between behaviours in the tree. blackboards[6]

A.3 Testing

Here are the steps that you can follow in order to run a demo of pytree in HARMONI. This demo show how work a small three with just two behaviours: microphone and stt.

1. Create a valid google credential of using stt service: here[7]

[4] <https://py-trees.readthedocs.io/en/devel/behaviours.html>
[5] <https://py-trees.readthedocs.io/en/devel/composites.html#composites>
[6] <https://py-trees.readthedocs.io/en/devel/blackboards.html#blackboards>
[7] <https://cloud.google.com/speech-to-text/docs/before-you-begin#setting_up_your_google_cloud_platform_project>

2. Run the following command in the container to start the demo:

```
roslaunch harmoni_pytree mic_and_stt.launch
```

You will see some information about the structure of the tree and the way in which pytree engines tick the tree. Some blackboards are used to show the results of the leaves and in particular in backboards stt/result shows the result of the speech to text services.

Appendix B

Imageai

ImageAI is a python library built to empower developers to build applications and systems with self-contained Deep Learning and Computer Vision capabilities.
Now in HARMONI you can find one among the several services that ImageAI provides: stream detection.

In particular we created two different services

- the first one use the state of art model for object detection yoloV3
- the other one use a custom model trained on the top of yoloV3

B.1 Setup

In order to use ImageAI import the python library by running the following commands:

“‘

$ pip install imageai –upgrade

$ pip install tensorflow==2.4.0 “‘

(recommended) For more information follow the official documentation: here[1]

[1] <https://github.com/OlafenwaMoses/ImageAI#dependencies>

B.2 Usage

So as to use plain yoloV3 model you have to do nothing but download the **yoloV3** model here[2] and then add it in the container in following path: `HARMONI/harmoni_detectors/harmoni_imageai/src/[PUT HERE]`.

As far custom models are concerned, there are some steps that have to be followed:

1. Create your own model following the steps in the documentation of ImageAI: here[3].
   At the end of this process you will obtain 2 files: a model.h5 and a config.json.
2. Put these two files in the following path `HARMONI/harmoni_detectors/harmoni_imageai/src/[PUT HERE]` in the container. Be sure that names coincide with the ones in `custom_configuration.yaml` file under `harmoni_imageai` folder.

Full documentation of ImageAI: here[4]

B.3 Testing

Here are the steps that you can follow in order to run the ImageAI service in HARMONI.

YoloV3

Run the following commands in order to run camera service and yolo service in two different terminals:

“‘

roslaunch harmoni*sensors camera*service.launch

roslaunch harmoni*imageai yolo*service.launch “‘

Camera will open and one frame will be captured and put as input of the yolov3 model. Once the output of the imageai service arrives another frame will be captured and again used as input for yolov3

[2] <https://github.com/OlafenwaMoses/ImageAI/releases/download/1.0/yolo.h5>

[3] <https://github.com/OlafenwaMoses/ImageAI/blob/master/imageai/Detection/Custom/CUSTOMDETECTIONTRAINING.md>

[4] <https://github.com/OlafenwaMoses/ImageAI#readme>

model and so on.

Custom yolo

Run the following commands in order to run camera service and custom yolo service in two different terminals:

“‘

roslaunch harmoni*sensors camera*service.launch

roslaunch harmoni*imageai custom*service.launch “‘

Camera will open and one frame will be captured and put as input of the custom model. Once the output of the imageai service arrives another frame will be captured and again used as input for custom model and so on.

Bibliography

[1] Neurodevelopmental disorders. https://ourworldindata.org/neurodevelopmental-disorders.

[2] Mindds - maximising impact of research in neurodevelopmental disorders. https://mindds.eu/mindds/.

[3] National Institute on Deafness and Other Communication Disorders. Autism spectrum disorder: Communication problems in children. *NIDCD Fact Sheet — Voice, Speech, and Language*, 2020.

[4] Felippe Sartorato, Leon Przybylowski, and Diana K.Sarko. Improving therapeutic outcomes in autism spectrum disorders: Enhancing social communication and sensory processing through the use of interactive robots. *Journal of Psychiatric Research*, pages 1–11, 2017.

[5] Real time computing. https://en.wikipedia.org/wiki/Real-time_computing.

[6] Behaviours tree wikipedia. https://en.wikipedia.org/wiki/Behavior_tree_(artificial_intelligence,_robotics_and_control).

[7] American Psychiatric Association. *Diagnostic and statistical manual of mental disorders*. 5th edition, 2013.

[8] Health. Neurodevelopmental disorders. *America's Children and the Environment*, 2015.

[9] Aarnoudse-Moens CS, Weisglas-Kuperus N, van Goudoever JB, and Oosterlaan J. Meta-analysis of neurobehavioral outcomes in very preterm and/or very low birth weight children. *Pediatrics.*, pages 717–728, 2009.

[10] Banerjee TD, Middleton F, and Faraone SV. Environmental risk factors for attention-deficit hyperactivity disorder. *Acta Paediatr.*, pages 1269–1274, 2007.

[11] Bhutta AT, Cleves MA, Casey PH, Cradock MM, and Anand KJ. Cognitive and behavioral outcomes of school-aged children who were born preterm: a meta-analysis. *JAMA.*, pages 728–737, 2002.

[12] Herrmann M, King K, and Weitzman M. Prenatal tobacco smoke and postnatal secondhand smoke exposure and child neurodevelopment. *Curr Opin Pediatr.*, pages 184–190., 2008.

[13] Richard E Behrman, Adrienne Stith Butler. Institute of Medicine (US) Committee on Understanding Premature Birth, and Assuring Healthy Outcomes. *Preterm Birth: Causes, Consequences, and Prevention.* 2007.

[14] Linnet KM, Dalsgaard S, Obel C, Wisborg K, Henriksen TB, Rodriguez A, Kotimaa A, Moilanen I, Thomsen PH, Olsen J, and Jarvelin MR. Maternal lifestyle factors in pregnancy risk of attention deficit hyperactivity disorder and associated behaviors: review of the current evidence. *Am J Psychiatry*, 2003.

[15] J.T Nigg. What causes adhd? understanding what goes wrong and why. *New York: The Guilford Press.*, 2006.

[16] Weiss B and Bellinger DC. Social ecology of children's vulnerability to environmental pollutants. *Environ Health Perspect.*, 2006.

[17] Everything you need to know about autism spectrum disorder (asd). https://www.healthline.com/health/autism.

[18] Autism treatment guide. https://www.healthline.com/health/autism-treatment#cognitive-behavioral-therapy.

[19] Early intervention makes a huge difference for autistic children. https://myasdf.org/media-center/articles/early-intervention-makes-a-huge-difference-for-autistic-children/.

[20] Gernsbacher MA. Dawson M. Effectiveness of intensive autism programmes. *Lancet*, 2011.

[21] J. J. Wheeler, M. R. Mayton, and Stacy Carter. *Methods for teaching students with autism spectrum disorders: Evidence-based practices.* Pearson, 2014.

[22] Autism q and a: What is applied behavior analysis? https://vcuautismcenter.org/resources/factsheets/printView.cfm/982.

[23] Michael A Goodrich and Alan C. Schultz. Human robot interaction : a survey. *Foundations and Trends® in Human-Computer Interaction*, 2007.

[24] Interaction design foundation. *The Encyclopedia of Human-Computer Interaction.* 2nd edition.

[25] J. Scholtz, M. Theofanos, and B. Antonishek. Theory and evaluation of human robot interactions. 36th International Conference on Systems Sciences, Hawaii: IEEE, 2002.

[26] A. Billard, B. Robins, J. Nadel, and K. Dautenhah. Building robota, a mini-humanoid robot for the rehabilitation of children with autism. *RESNA Assistive Technology Journal*, 2006.

[27] P. Stribling B. Robins, P. Dickerson and K. Dautenhahn. Robot-mediated joint attention in children with autism: A case study in robot-human interaction. *Interaction Studies, vol. 5, no. 2*, page 161–198, 2004.

[28] I. Werry, K. Dautenhahn, B. Ogden, and W. Harwin. Can social interaction skills be taught by a social agent? the role of a robotic mediator in autism therapy. 2001.

[29] Momotaz Begum, Richard W. Serna, and Holly A. Yanco. Are robots ready to deliver autism interventions? a comprehensive review. *International Journal of Social Robotics volume 8*, page 157–181, 2016.

[30] Laurie A Dickstein-Fischer, Darlene E Crone-Todd, Ian M Chapman, Ayesha T Fathima, and Gregory S Fischer. Socially assistive robots: current status and future prospects for autism interventions. *Innovation and Entrepreneurship in Health*, pages 15–25, 2018.

[31] Huijnen CAGJ, Lexis MAS, Jansens R, and de Witte LP. How to implement robots in interventions for children with autism? a co-creation study involving people with autism, parents and professionals. *J Autism Dev Disord.*, 2017.

[32] Caroline L. van Straten, Iris Smeekens, Emilia Barakova, Jeffrey Glennon, Jan Buitelaar, and Aoju Chen. Effects of robots' intonation and bodily appearance on robot-mediated communicative treatment outcomes for children with autism spectrum disorder. *Pers Ubiquit Comput 22*, page 379–390, 2018.

[33] Robins B, Dautenhahn K, and Dubowski J. Does appearance matter in the interaction of children with autism with a humanoid robot? *Interaction Studies, Volume 7, Issue 3*, pages 479 – 512, 2006.

[34] Scassellati B. How social robots will help us to diagnose, treat, and understand autism. 2007.

[35] Ben Robins, Nuno Otero, Ester Ferrari, and Kerstin Dautenhahn. Eliciting requirements for a robotic toy for children with autism - results from user panels. *RO-MAN 2007 - The 16th IEEE International Symposium on Robot and Human Interactive Communication*, pages 101–106, 2007.

[36] Daniel J. Ricks and Mark B. Colton. Trends and considerations in robot-assisted autism therapy. *2010 IEEE International Conference on Robotics and Automation*, pages 4354–4359, 2010.

[37] Diehl JJ, Schmitt LM, Villano M, and Crowell CR. The clinical use of robots for individuals with autism spectrum disorders: A critical review. *Res Autism Spectr Disord.*, pages 249–262, 2012.

[38] David Feil-Seifer and Maja J. Matarić. Defining socially assistive robotics. *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pages 465–468, 2005.

[39] Feil-Seifer, David, Matarić, and Maja J. Toward socially assistive robotics for augmenting interventions for children with autism spectrum disorders. In Oussama Khatib, Vijay Kumar, and George J. Pappas, editors, *Experimental Robotics*, pages 201–210, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[40] Carole A Kubota and Roger G Olstad. Effects of novelty-reducing preparation on exploratory behavior and cognitive learning in a science museum setting. *Journal of research in Science Teaching 28, 3*, page 225–234, 1991.

[41] Timothy W. Bickmore and Rosalind W. Picard. Establishing and maintaining long-term human-computer relationships. *ACM Trans. Comput. Hum. Interact.*, 12:293–327, 2005.

[42] Ilias Katsanis and Vassilis C. Moulianitis. *Criteria for the Design and Application of Socially Assistive Robots in Interventions for Children with Autism.* 2020.

[43] Huijnen CAGJ, Lexis MAS, Jansens R, and de Witte LP. Roles, strengths and challenges of using robots in interventions for children with autism spectrum disorder (asd). *Journal of autism and developmental disorders*, pages 11–21, 2019.

[44] Ji, Sang Hoon, Su Jeong, Cho, and Hye-Kyung. Design of emotional conversations with a child for a role playing robot. page 73–74, New York, NY, USA, 2015. Association for Computing Machinery.

[45] Stephan and Achim. Empathy for artificial agents. *International Journal of Social Robotics*, 7(1):111–116, 2015.

[46] Nicole Giullian, Daniel J. Ricks, John Alan Atherton, Mark B. Colton, Michael A. Goodrich, and Bonnie Brinton. Detailed requirements for robots in autism therapy. *2010 IEEE International Conference on Systems, Man and Cybernetics*, pages 2595–2602, 2010.

[47] Clabaugh, C., and Matarić. Escaping oz: Autonomy in socially assistive robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, pages 33–61, 2019.

[48] Matarić, M.J., and Scassellati. *Socially assistive robotics*. 2016.

[49] Wainer, Robins, Amirabdollahian, and Dautenhahn. Using the humanoid robot kaspar to autonomously play triadic games and facilitate collaborative play among children with autism. *IEEE Transactions on Autonomous Mental Development*, 6(3):183–199, 2014.

[50] Qtrobot offical website. https://luxai.com/robot-for-teaching-children-with-autism-at-home/.

[51] Sheila Christopher. Picture cue cards-a tool in enhancing communication in children with autism. 09 2013.

[52] Spitale Micol, Birmingham Chris, Swan, R. Michael, and Maja J Matarić. Composing harmoni: An open-source tool for human and robot modular open interaction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[53] Behaviours tree implementation methods. https://github.com/BehaviorTree/awesome-behavior-trees.

[54] Harmoni repository, thesis branch. https://github.com/micolspitale93/HARMONI/tree/feature/pytree.

[55] Pytree documentation. https://py-trees.readthedocs.io/en/devel/index.html.

[56] Imageai repository. https://github.com/OlafenwaMoses/ImageAI.

[57] Yolov3 architecture. https://dev.to/afrozchakure/all-you-need-to-know-about-yolo-v3-you-only-look-once-e4m.

[58] Confusion matrix and others parameters. https://towardsdatascience.com.

[59] Amazon lex. https://docs.aws.amazon.com/lex/.

[60] Google cloud speech. https://cloud.google.com/speech-to-text.

[61] Wemos chipset. https://www.wemos.cc/en/latest/d1/d1_mini.html.

[62] Firebase. https://console.firebase.google.com/.