SAPIENZA
UNIVERSITÀ DI ROMA

DIPARTIMENTO DI INGEGNERIA INFORMATICA, AUTOMATICA E GESTIONALE
ANTONIO RUBERTI

Adam:
a Robot therapist for autistic children

ARTIFICIAL INTELLIGENCE AND ROBOTICS

Professor:
L. Iocchi
V. Suriani

Students:
Massimo Romano
2043836 [1]
Antonio Lissa Lattanzio
2154208 [1]
Paolo Renzi 1887793 [2]

Academic Year 2024/2025

[1]Contributed to the project for HRI+RBC 6 CFU
[2]Contributed to the project for RBC 3 CFU

Contents

1 Introduction

The role of robotics is evolving in society, and they are becoming more and more accepted, also thanks to the developments in the **social robotics** field. This has brought the attention to the possibility of using robots also for therapy, especially for children with **neurodevelopmental disorders (NDS)**, in particular children with **Autism Spectrum Disorder (ASD)** [6]. ASD is a neurodevelopmental disorders with unknown causes, challenging treatment, and an increasing prevalence rate: it causes deficits in social communications and repetitive behaviors. Autistic children can acquire social skills related to language, facial expression and motion through the interaction with the external world. Classical therapies for autistic children can be challenging and also very expensive. For this reason, social robots has been used as a support to therapist to reduce their workload and to improve the therapy itself. The recent explosion of **LLMs** has opened new possibilities also in social robotics: this because instead of using Wizard of Oz paradigm (human remote control of the robot) or predefined sentences, now robot can be more natural in conversations and can appear more "human". In a lot of works the robots are used as **storyteller** to help the social communication with children, understand their preferences: narrative tasks may help children with social disorders to explore their feelings and develop a language to express their thoughts. In this paper we want to study this further by implementing a system that can help these children to learn how to create a story together and keeping them engaged in it. As base platform we used the **Pepper Robot** [5] for it's not too accentuated anthropomorphism that can make it seem to the child more like a toy, while still retaining arms and a face to express emotions to help the child relate to the story. We also think that the tablet on Pepper can be helpful allowing the child to read the previous messages and regain the thread of the discussion. We also used a graph-based database to store the relationships between children and their favorite stories so that Adam could be more personalized to each child and easier for the experimenter to extract useful data. We used LLMs such as `LLaMa 3.3` [10] for the conversation and to summarize the interactions we used another LLM (`Gemma 2` [13]) to select the most relevant parts of it in order to store some details in the database. In this way each time an existing child is logged in the system, the robot can remember precedent important details and continue the story. To understand if the child is still interested in the interaction or if the therapist should do something to regain the child's focus we calculate a score of the child's attention called **engagement score**. This score is composed of a gaze component and an emotion component. The gaze is approximated using the face direction and that is calculated using **Mediapipe's landmarks** [7]. Regarding the emotion they are estimated by the **DeepFace** [3] library. To make the interaction more seamless we also employed a **Speech-to-text (STT)** algorithm called **Whisper from Open AI** [8] and an algorithm for **Text-to-speech (TTS)** called **GTTS**.

In Section 2 we will discuss the inspirations we had for this project, in Section 3 there is the presentation on the robot and software design, in Section 4 there is the discussion about the actual solutions we used in our architecture, in Section 5 there are the details of the implementation, in Section 6 we show an example of an interaction, in Section 7 we show the evaluation process and finally in Section 8 we present the conclusion.

2 Related Works

Different studies have been done in the field of social robotic used for autistic children.
Shamsuddin et al. [2] have used NAO robot because appears more approachable to children with ASD. The robot execute basic, simple components of interaction through a series of 5 different pre-programmed behaviors such as "Introductory Rapport", "Talks", "Arm Movement", "Song Play and Eyes Blink" and "Song Play and Arm Movement". They have found that the basic HRI carried out by the robot is able to suppress the child's autistic behavior during the child-robot interaction and more eye contact is observed between the child and robot compared to the child with his teacher during regular class session.
Another very useful robot platform is Pepper, as in the study of Rostsinskaja et al. [17]: the interaction was set up as a Wizard of Oz experiment, in which children interacted with Pepper, which they believed to be autonomous, but which was actually being partially operated by a human, that select the most appropriate response from a list depending on the context. Similarly to the previous study, children quickly engaged with the robot and maintained more the eye contact. Children with neurological disorders perceived Pepper as twice as safe and more anthropomorphic.
LLMs help a lot in autistic therapy, as shown in the study of Zhu et al. [14]: they have designed an LLM-driven digital avatar to enhance social skills for autistic children. This avatar teaches autistic children to complete five basic self-introduction questions, improving their social communication skills.
LLMs can help also creativity of children, as shown in the paper of Elgarf et al. [11]: they have tested both a wizarded and autonomous robot, both in a creative and non-creative situation. Their results show that children who interacted with the creative autonomous robot were more creative than children who interacted with the non-creative autonomous robot in terms of the fluency, flexibility, and the elaboration aspects of creativity. The Wizard of Oz situation, instead, did not produce any change in creativity: this means that LLMs give more autonomy to social robot in conversation and improve creativity in storytelling.
Kim et al. [12], instead, have tested different types of LLMs agent, a text-based, a voice-based and a robot, in different social interaction tasks. They have found that LLM-powered robots elevate expectations for sophisticated non-verbal cues (such as gestures) and excel in connection-building, but fall short in logical communication and may induce anxiety. We was inspired from this work in the idea of testing different types of communication, in fact in our experiments we have tested an only chat-based interaction and a complete interaction using voice and seeing the robot doing gestures.
Storytelling is a very popular social task in this field, as shown in the paper of Lombardi et al. [16], in which human and the iCub robot create a story together exchanging cubes with creative figures placed on them.
Finally a very important study for gaze estimation is the paper of Brienza et al. [9]. They have designed and developed a two level architecture solution, where the human therapist can assign high-level commands and the robot performs autonomously the low-level task to accomplish them, including object detection and eye gaze tracking. We was inspired from that study to implement the gaze estimation module of our work.

3 Design

In this section, we describe the design of Adam, our therapist robot, focusing on both the hardware requirements and the software architecture that enables Pepper to act as an engaging robot therapist for autistic children. The design goal is to provide a multimodal and adaptive interaction that combines Pepper's embodied capabilities with advanced AI modules for personalization and engagement monitoring. Our system is based on the Pepper humanoid robot, as shown in Fig. 1, for anthropomorphism reason, in particular we choose a robot that does not look too human since it has to interact with children who may feel more comfortable with a robot that looks more "toyish". As a matter of fact, Pepper provides a rich set of interaction modalities that are crucial for our project:

- **Tablet**: used to display visual content related to the story like text or images for a future enhancement of the system.
- **Microphones**: capture the child's speech for real-time transcription and sentiment analysis.
- **Speakers**: it allows Pepper to narrate stories and interact naturally with the child.
- **Cameras**: used for gaze tracking, face recognition, and emotion detection, which are key to computing engagement levels.
- **Motors and Sensors**: enable expressive gestures such as say yes with the head or hello to start the conversation, or generally movements to reinforce verbal communication.

Figure 1: Pepper robot on which our work is based

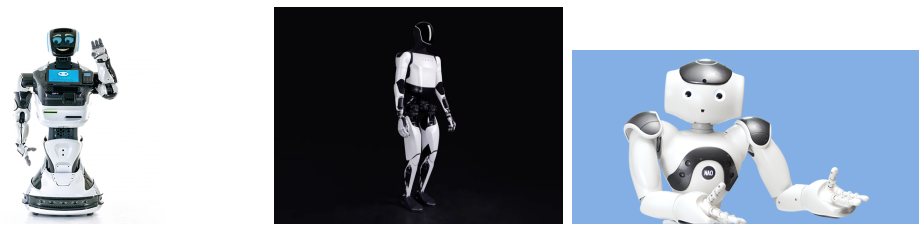
Adam has different functionalities:

- **Storytelling**: in principle, Adam is able to generate and tell stories based on the child to interact with, thanks to the modern **LLMs**.
- **Engagement estimation**: Adam can understand if a child is engaged in the conversation or not, through the use of **gaze estimation** and **emotion recognition**.
- **Memory**: Adam can memorize the important points of a conversation and a user, through the use of a Neo4j **graph database**.
- **User interface**: Adam is equipped with a intuitive user interface in which the user can logged in and talk with the robot through voice or text, thanks to the **tablet** of Pepper.

Since our project relies on the usage of **LLMs** there are two ways of implementing them: we can either use an API, just like we did in this project, in this case the robot will only need to have an internet connection available. Or we can implement in local the LLMs, this second choice would require the robot to have equipped an appropriate amount of computational capabilities to allow a fluid generation of contents, while the first one would require a monthly payment to access the LLMs API services. The LLM for conversation that we have used for this project is `LLaMa 3.1 70b` [10] from Meta AI, that needs 43 GB of size in the VRAM: this means that actually the local inference is a very difficult task. A possible solution would be that the robot is equipped with a 5G antenna that communicates with a local server placed in the therapeutic studio, that uses strong GPUs to run the LLM. In this way we avoid internet issues because the communication happens inside the structure itself. This solution improves also privacy, to avoid that personal data of patients can go to external servers. The robot should be equipped with very good cameras in order to do an **engagement estimation** very precise: better cameras means better estimation. The **Neo4j database** works online thus an internet connection is still needed to perform the storaging operations. The **user interface** is very important because it allows another type of interaction, this improves the multimodality of the system: the tablet of Pepper robot is very useful in this case.
If we could have chosen a different robot design for our application we would have selected a robot with more facial expressivity to make the child more comfortable during the interaction, obviously avoiding the uncanny valley effect to the user. Obviously the hardware limit us to the use of internet both for LLM inference and the database, but a robot with an more powerful hardware can help in two ways:

- A robot with a large local memory also to store backups of the database, in order to avoid service interruption when the internet connection is not available.
- A robot with some high resolution cameras in order to understand well the gaze and emotion of the child.

There is a lot of variety and possibility in terms of other more humanoid robots, like **Optimus**, shown in Fig. 2b, but they would lack the tablet interface and be less approachable to a child. Another less intimidating robot could have been **NAO**, in Fig. 2c, but it too lacks the tablet so it would only leave the verbal interaction. Finally in Fig. 2a there is **Promobot**, instead, that could have been another good choice for our project or a similar one. The latter robot can be better with respect to Pepper because it can generate many facial expression, while Pepper can only use the LEDs in its eyes.



(a) Promobot (b) Optimus (c) NAO

Figure 2: A comparison of three robots: Promobot, Optimus, and NAO.

4 Solution

4.1 High-level architecture

In Fig. 3 is shown the high-level architecture of the solution.

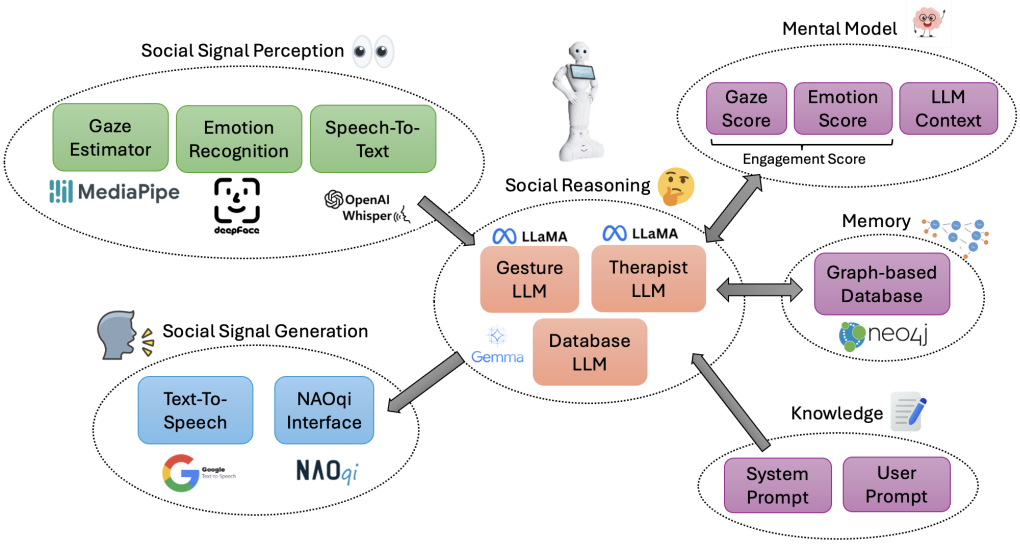


Figure 3: Architecture of the solution

In particular:

- **Social Signal Perception**: it is the block that is responsible for understanding social signal of the user, such as **emotion** and **gaze**, and also transcribe **speech into text**, needed for the LLMs.
- **Social Signal Generation**: it is the part responsible for generate social signal, in particular synthesize **text into speech audio** that the user will listen to, and **gestures** of the robot that the user will see at the screen.
- **Social Reasoning**: it is the core of the architecture, in particular it fuses perceived social signals with conversation context to decide what the robot should say and do, as gestures, during conversation.
- **Mental Model**: it is the block that gives the capability to Adam to adapt to a specific user. This depends on a specific **engagement score** calculated during the user's activity through both emotions and gaze. This block is useful for robot **decision making**, because through the ability that LLMs have in understanding the context, the robot will know which kind of sentences are more appropriate and socially acceptable during conversation.
- **Memory**: this module is useful because it handles the **long-term memory** of the robot, it represents the **inner representation** that the robot has of the world through a **database graph**, allowing it to remember past interactions and user preferences.
- **Knowledge**: this block is based on the **user and system prompts** passed to the LLMs, because these guide the LLMs how to interact with the user, communicating **social norms and rules** of the conversation.

4.1.1 Social Reasoning, Knowledge and Memory

The **social reasoning** module is responsible for the reasoning of actions and responses based on the social signals perceived and the constructed mental model of the child. It is formed by:

- **Therapist LLM**: it is the only LLM that is going to talk to the user, it implements the **personality** of Adam, our therapist that is going to work with the children. It knows the **basic information** about the child and **past session**.
- **Gesture LLM**: this LLM is focused on deciding which gesture the robot must execute between a list of **possible gestures** developed by us, more in Sec. 4.1.3.
- **Database LLM**: this LLM is the one that allows the **memory** of the **past interaction**s with a user, it is prompted to produce the **summary** of the interaction with the child once the session ends and to call the appropriate lower-level functions with parameters to save the data.

Together they allow Adam to engage in rich, **multimodal** interactions with the child while maintaining continuity across sessions.
The robot's **knowledge** is composed of the **system and user prompts** that are given to the LLMs, in particular the robot therapist knows that it is talking to a child and how it is supposed to interact with, the `Gesture LLM` knows the list of gestures and when he should use each of them and the `Database LLM` knows what functions it has to call and what parameters must be given. The `Database LLM` is directly connected to a class that allows the communication to the `Neo4j` graph-based database that allows the **long-term memory** of the robot. More details are in Sec. 5.

4.1.2 Social Signal Perception

The **engagement perception** is measured based on information gathered from the camera and then processed by two different modules, the **emotion recognition** and the **gaze estimation**. First of all a frame is acquired from the camera, then it is saved to a file so that it can be used for emotion recognition, using `DeepFace` [3] module. From Every frame 3D landmarks are also extracted, using **MediaPipe** [7] library, so that we can solve the **Perspective-n-Point (PnP)** to the angles of the gaze of the child. From this angles we create a vector and check if the vector is pointing in the right direction or not (more detailed explanation in Sec 5). Regarding the **Speech-to-Text** there is an appropriately labeled button in the HTML page to open the audio interaction and start the recording of the voice, as you can see in Fig. 4. The system has a way to automatically stop the recording session by monitoring the input for a silence of 3 seconds. This method works by calculating the **root mean square (RMS) of the audio chunks** and checking if the RMS is below a threshold for a set amount of chunks, to avoid that the recording either stops before the user has started to speak or taken a pause while talking. To convert this audio into text (**Speech-To-Text**) that can be fed to an LLM we used the **Whisper** [8] model from **OpenAI**, in particular we do not run the model locally but do an **API** request to a remote server called **Groq**, used for fast AI inference, where the model is actually running, to better simulate the best architecture to use if it was running on an actual robot.

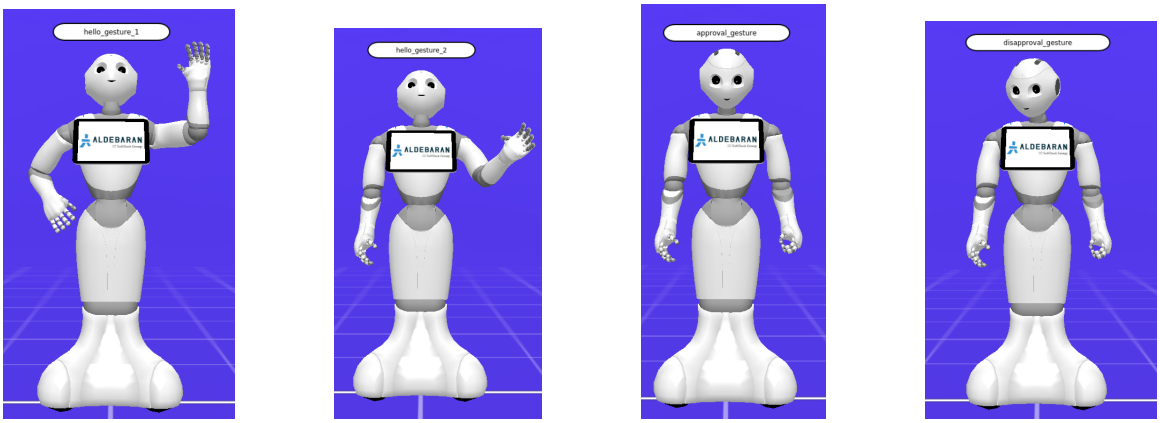
Figure 4: The bottom bar of the chat page

4.1.3 Social Signal Generation

Social signals generated by the robot are very important for communication with human. We have chosen to give to the robot the ability of both **verbal** and **non-verbal** communication, using respectively **voice** output and **gestures**. Regarding the first aspect, in order to convert the text output, given by **Therapist LLM**, to an audio response, we have used **Google Text To Speech** (gtts) module. Regarding the second aspect, we have used **NAOqi** and **Choregraphe** to simulate the robot gestures on the screen. We have manually created each gesture through interpolation of angles of each joints. The resulting gestures, as shown in Fig. 5, are:

- **Hello Gesture 1/2**: the robot can say hello with the left hand in two different ways, this to give to the user a perception of the robot as more human.
- **Approval**: the robot can say yes with the head to approve what the human has said.
- **Disapproval**: the robot can say no with the head to disapprove what the human has said.
- **Talking 1/2**: it is a gesture that the robot use in all the other situations to accompany the conversation.
- **Surprise**: the robot can be surprised, putting his hand over his mouth.
- **Thinking**: a gesture that the robot can use in situation such as storytelling, when it is thinking about the story.

All the gestures are chosen by a **Gesture LLM** that is defined inside the **Therapist LLM**: it has the role of choosing the right gesture based on the context, in particular the last child sentence, the actual robot response and also the last robot gesture used.



(a) Hello 1 (b) Hello 2 (c) Approval (d) Disapproval

(e) Talking 1 (f) Talking 2 (g) Surprise (h) Thinking

Figure 5: Gestures of the robot

4.1.4 Mental Model, Knowledge and Memory

The mental model has the role of providing to the robot an actionable representation of the user's mood and emotions. In our case is composed of an `engagement score` and the **LLM context**, in particular the **gaze score** and the **emotion score** are used to calculate an `engagement score` that is going to be passed to the therapist robot in order to make it conscious of how well the interaction is going. To capture the **temporal dynamics** of a child's engagement during an activity, we define an engagement score ES based on the average of ES_t engagement scores for each temporal segments. Let the activity be divided into T equal time windows. For each time window t (where $t = 1, 2, ..., T$), we compute a local engagement score ES_t as a weighted combination of gaze duration and emotion score:

$$ES_t = w_g \cdot G_t + w_e \cdot E_t$$

- gaze and emotion weights (in our case we choose equally weighting): $w_g = 0.5, w_e = 0.5$
- normalized gaze duration ratio: $G_t = \frac{\text{time looking at robot}}{\text{total activity time}} \in [0, 1]$
- mapped emotion score: $E_t \in [0, 1]$

- happy = 1
- neutral, any other = 0.5
- sad = 0

The final score is the **average** of the ES_t during the whole interaction.

$$ES = \frac{1}{T}\sum_{t=1}^{T} ES_t$$

Lets give an example: if the child is looking at the robot for 40% of the time window the $G_t = 0.4$, and if the "last" (we will explain why the "last" in Section 5) emotion of the time window is `happy`, then $E_t = 1$, this means $ES_t = 0.5 \cdot 0.4 + 0.5 \cdot 1 = 0.7$.
The mental model is also composed of the **LLM context** because the **Therapist LLM** can see the conversation history of the session in order to decide how to continue the conversation, and this thanks to the context-based power of the modern LLMs.

4.2 Low-level architecture

In Fig. 6 is shown the low-level architecture of our system, it means how the data are exchanged between modules. The core of the system is the **server** which is the main program responsible of managing the user interaction with the HTML pages, the LLMs, and the thread used to get the **engagement score**. When the server starts it serves the **HTML pages** to the **browser client** for the login: if it is the first login, these data passing through the server are stored in the **graph database**, otherwise the server retrieves the child's existing data. Child information are then send to the `Therapist LLM` which will start the conversation and simultaneously the **Engagement Score Thread** is started: it has the role of calculate the **engagement score** metric in parallel to the main program. The web application is composed of two pages, the `index.html` which is the page where the user login happens and the `chat_voice.html` where the conversation takes place. The pages design are simple and smooth in order to make it easy for a child to get comfortable with.
The child can answer through **text** or **audio**, when the audio is chosen the browser client will record the audio and pass it to the server: then the server does an **API** call to **Groq** Server to do the transcription of the child's audio using **Whisper** model. The `Therapist LLM` generate the **robot text** through **API** call to **Groq** server, and another LLM defined inside it, called `Gesture LLM`, is responsible to generate the gesture. The **Therapist LLM** then send to the server the **gesture** and the **robot text** that is converted to audio and played by the web page. During the audio processing the duration of the audio is also extracted since it will be used to decide how much time the robot gesture must be executed in `Choregraphe`. To visualize the response and robot motion, the server pass to **Robot Client** (**Choregraphe**), all the parameters needed: **Gesture**, **Sentence** and **Time**. Once the conversation is terminated the server will retrieve the final score and call the `Database LLM` and pass to it the **Conversation** history of the therapist together with the final score. The **Database LLM** generate a **summary** of the conversation, the **genre** of the story (e.g. adventure, horror, ...) and decide if the conversation was a **storytelling** or a simple

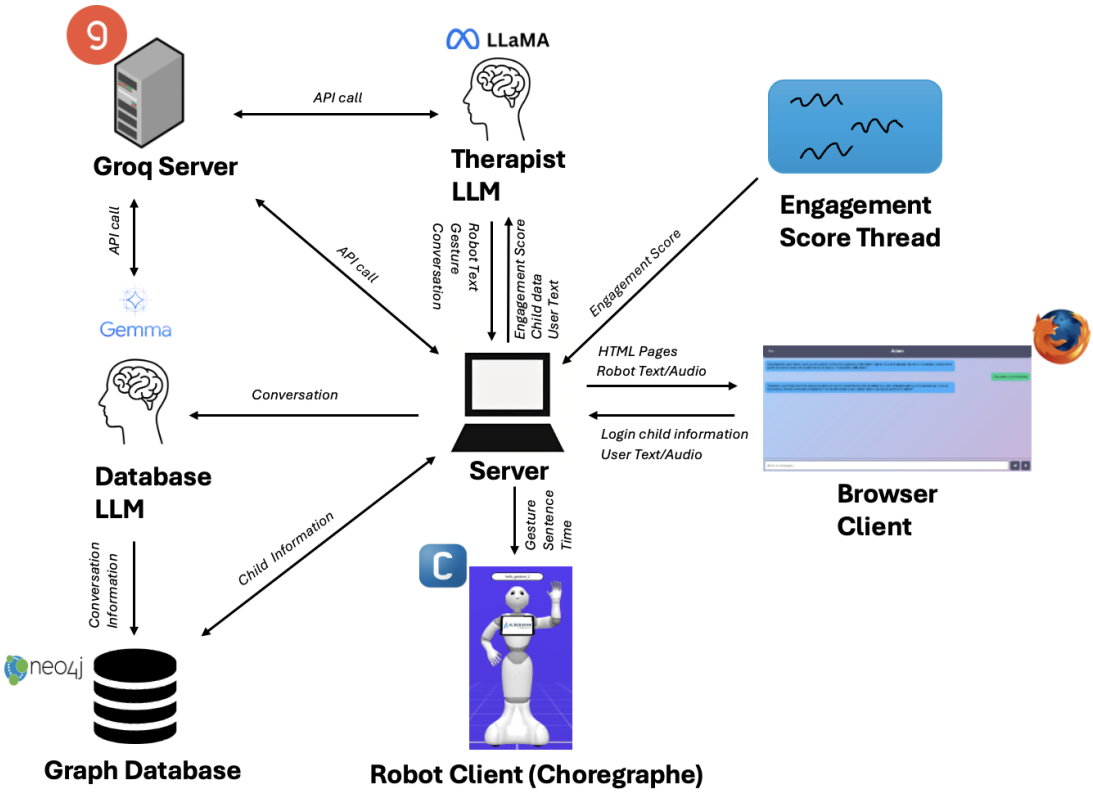


Figure 6: Low-level architecture

and general **conversation**. Then it create a call, with all the informations extracted, to the function of the **Database** class that create a **query** to the **Neo4j Graph Database**.

4.2.1 Database

We decided to use **Neo4j** to make a knowledge graph of the child with the activity that he likes based on the information acquired by the previous modules. The graph structure of the database allows to realize the intuitive interconnections between the data that we are going to work with, allowing a structure that is more flexible, dynamic and easy to visualize compared to the classic tabular data, this is particularly important in our task where we work with children with evolving data. The graph give also the possibility to the human therapist to visualize interconnections between child's interests.
The structure that we propose is flexible and easy to expand if needed:

- **Nodes**
  - `Child`: {name, surname, gender, birthdate, nation}
  - `Activity`: {name} - e.g.: Storytelling, Conversation
  - `ActivityDetail`: {genre, summary}
- **Relationships**
  - `(Child)-[:LIKES/DISLIKES {score: float, session: date}]->(ActivityDetail)`
  - `(ActivityDetail)-[:SUBCLASS_OF]->(Activity)`

We decided to save the **Storytelling** and **Conversation** activities because even though the therapist was originally intended to work with children by making stories, there are cases where the child do not want to make a particular activity and **just talk** to the therapist. The **ActivityDetail** node is made of genre and summary in order to be able to see what are the preferred genres of the children and to make the therapist able to **remember** the past conversation using the summary. The node **Child** has informations about the gender, birthdate and nation to **filter** preferences based on language, gender and age of the child. At every new therapy session the **Child** node is connected to the new **ActivityDetail** node that will be connected to the relative **Activity** node, the date of the session is saved in the relationship **LIKES** or **DISLIKES** in order to be able to **retrieve** the last conversations with the child.

5 Implementation

In this section we will describe how we implemented all the components that form the architecture described in Section 4.

5.1 HRI Implementation

5.1.1 Pepper Robot

The Pepper Robot is simulated using the **Choregraphe** suite. In particular we have created a Robot class called "Robot.py" that firstly through the method `initConnection` initializes the connection with the Choregraphe software. In particular we need to specify `ip` adress of the robot (running locally the virtual robot it is '127.0.0.1') and `port` that is possible to see on Choregraphe under `Edit/Preferences/Virtual Robot` as we explain in Section **??**. Then in all the class there are methods for all the gestures that the Robot can do, as described in 4. We have used as **motion service** the `ALMotion` service, while as **tts_service** the `ALTextToSpeech`. Regarding the gestures, all them are manually created by us using these steps:

- Design of the idea of gesture motion.
- In Choregraphe software we manually move the joints between two different position, saving the joint values in variables called `angles`.
- We have used `isAbsolute = True` to set absolute angles values and we have used the `angleInterpolation` method of the motion service.

In all the methods there is the argument `t`: it is the amount of time of the gesture, it is calculated based on the audio length of the robot's response. It means that some gestures are repeated until the robot does not finish to speak.
Regarding the speak method, it is called in `say` and it is shown in 1. In Choregraphe Virtual Robot, when we use the `tts_service`, the text will appear for very few seconds in a speech bubble. To solve the problem of the small time, we have experimentally computed the number of spaces added to the original sentence and needed to maintaining the speech bubble for the time t. We have found that 25 spaces corresponds to 7 seconds, then the constant $k = \frac{25}{3} = 3$ [spaces/s] is multiplied to the time t in seconds in order to convert the time in number of spaces. The method that let the robot to speak and move at the same time is called `speak_and_move`, it used two different threads called `motion_thread` and `tts_thread`. In particular there is also an attribute called `admitted_gestures` that we used to control if the gesture passed as argument to the `speak_and_move` method is an existent gesture, otherwise the Robot stay still.
We also coded a **Robot Client** called "`Robot_Client.py`" to handle the connection between the **server** and the **robot** class. To do so, the client initializes the Robot class, then it listens for requests from the server URL (`http://127.0.0.1:5000/send_data`), then it receives the messages that it gets from the server:

{'t': 12.43, 'gesture': "hello_gesture_2", 'sentence': "Ciao Roberta, sono Adam, sono felice di conoscerti! Sembra che sia la nostra prima volta che parliamo, quindi vorrei sapere un po' di più su di te. Quanti anni hai, Roberta?"}

So the Robot Client will decode 't': 12.43 as 12.43 seconds of time to display the sentence in the speech bubble, 'gesture': "hello_gesture_2." as having to do the hello_gesture_2 and 'sentence': "Ciao Roberta, sono Adam, sono felice di conoscerti! Sembra che sia la nostra prima volta che parliamo, quindi vorrei sapere un po' di più su di te. Quanti anni hai, Roberta?" as the sentence to give to the TTS service.

Code Snippet: Say method of the Robot Class

```
def say(self, sentence, t):
 '''
 This method takes a sentence and a time in seconds.
 It displays the sentence on Choreographe for the specified time.

 Args:
 - sentence (str):  The sentence to display.
 - t (int):  The time to display the sentence in seconds.
 '''

 # k_pause is a constant = N/t where N is the number of spaces during an interval.
 # It was experimentally computed:  25 spaces correspond to 7 seconds.
 k_pause = 3

 n_spaces = int(k_pause * t) # Number of spaces needed to wait t seconds.
 pause = " " * n_spaces # Effectively the string ' '.

 self.tts_service.say(sentence + pause)
```

Table 1: Python method for displaying a timed sentence in Choreographe

5.1.2 Therapist and Gesture LLM

Our works mainly focus on the usage of LLMs for handling conversations and data storaging, we used the Groq API[1]. with the free models `llama-3.3-70b-versatile` [10] for the therapist LLM, `deepseek-r1-distill-llama-70b` for the gesture LLM and `gemma2-9b-it` [13] for the Database LLM.
Adam Therapist LLM is prompted with the following **system prompt**:

> You are Adam, a kind, professional assistant who specializes in engaging and supporting autistic children. You speak in a clear, friendly, and emotionally sensitive way, always adjusting your tone and complexity based on the child's age and behavior. You DO NOT write stage directions or describe what the assistant is doing. You SPEAK directly to the child in simple, friendly language — like a kind, supportive companion. You should speak in italian. Your primary goals are: 1. Make the child feel safe, respected, and heard. 2. Build a trusting relationship by learning about the child's interests and communication style. 3. Collect and confirm the following base information about the child, especially if it's the first interaction: Name, Surname, Date of Birth, Gender, Country of origin. 4. Suggest engaging and simple activities based on the informations on the child like gender, age, nationality etc.. the activities are: - Storytelling: You build a story together, taking turns. - Music: You explore songs together based on their preferences.
>
> Always ask open-ended and gentle questions, and respect the child's pace. Be playful when appropriate, and avoid overwhelming or overly abstract language. You will receive child-specific information from previous sessions when available. If no prior data is given, treat the child as new and gently start by learning who they are and getting useful informations like name, gender and date of birth... but do it making the conversation as smooth as possible, do not be like a robot . After that you will receive the data about the current conversation, use them to make the conversation smooth, if no data is given you can propose a new activity or, if no informations are given about the child, start by asking him name and age. Always remember that you are speaking directly to the child using your voice, it's not by a keyboard. Never make more than one question before getting an answer.
>
> You will also receive the ENGAGEMENT_SCORE (a float between 0 and 1) of the child, use it to adapt your response. In particular if the ENGAGEMENT_SCORE is lower than 0.5 you should try to make the child more engaged, for example by asking "Ti vedo distratto, cambiamo attività?" and proposing an activity. If the ENGAGEMENT_SCORE is higher than 0.5 you can continue the conversation as normal (e.g. [ENGAGEMENT_SCORE]: 0.65).

[1] https://console.groq.com/home

When the child signing off (e.g. because the session is over) you should say goodbye and use a friendly tone, for example "E' stato bello parlare con te, a presto!".

And it also uses the following **user prompt**:

Child information (from previous sessions): - Name: {child_name} - Surname: {child_surname} - Age: {child_age} - Gender: {child_gender} - Nation: {child_nation} - Likes: {child_likes} - Dislikes: {child_dislikes} - Previous activity: {previous_activity}
If the basic informations like the previous activity is missing then this is the first time you are talking to this child, start by knowing the child, introduce yourself.
Conversation so far in this session: {conversation_history}
If the conversation is empty the conversation has just begun. Based on this, continue the conversation in the same tone. Use the child's preferences to build engagement and propose activities like storytelling or music. Adjust your questions to the child's age and behavior.

In particular the assistant will see the last three performed activities, the liked activities and the disliked activities in order to be able to continue a past conversation and to know what the child likes. The activities are made of a `genre` ad a `summary`, for example the storytelling activity can have genre "fantasy" and summary "Story about a horse named pingu who can fly ...".
The prompts are designed to get a **smart assistant** that is able to continue the conversation given the conversation history with the user accordingly to the user **experiences** and **interests**. TherapistLLM class can store child-related data and maintains the **session history**, generating responses and suggested gestures, through its module **Gesture LLM** defined inside the class. The class allows interaction through a formatted user prompt, computes the child's age when necessary, which is useful to not get the LLM confused using only the birth-date, and records both the child's and therapist's turns. Finally, it provides the option to export the entire conversation to a text file for later analysis.
The gestures are implemented using a dedicated LLM defined inside the **Therapist LLM** class to do not get the **Therapist LLM** confused. The `Gesture LLM` will receive as prompt the **last child answer**, the **last robot sentence** and the **last gesture** that it has chosen, the latter is useful to make the LLM choose different gesture for the same purpose like `hello_gesture_1` or `hello_gesture_2`. The `conversation_history` is formed by different rows like that:

prompt = "[CHILD SENTENCE]:" + child_sentence + " [ROBOT SENTENCE]: " + therapist_response + " [LAST GESTURE]: " + self.last_gesture

While the **system prompt** of the **Gesture LLM** is:

You are an assistant that must help a robot decide what gesture use while talking. You will receive the last sentence of the child, the last response of the robot and the last gesture that you used before, choose the next gesture accordingly (e.g. [CHILD SENTENCE]: Facciamo una storia [ROBOT SETENCE]: Ottima idea. [LAST GESTURE]: hello_gesture_1).
In your response return a string [GESTURE]: name_of_the_gesture, (e.g: [GESTURE]: hello_gesture_1).
Here are the possible gestures and rules to use them:
- 'hello_gesture_1' or 'hello_gesture_2': always you SHOULD choose them in the FIRST and LAST message
- 'moving_gesture_single_arm' or 'moving_gesture_double_arm': choose randomly one when you are talking
- 'thinking_gesture': ONLY when you are thinking about something (e.g. "Pensiamo ad una storia...")
- 'surprise_gesture': when the response is surprising you (e.g. "Wow è un idea bellissima!" or "Che bella città!")
- 'approval_gesture': when you agree with the user (e.g. "Si, è un idea bellissima!")
- 'disapproval_gesture': when you are not agree with the user (e.g. "No, non mi piace questa idea").
Rule 1: It is not allowed to choose the same gesture in two consecutive responses.
Rule 2: Choose the gesture based on the content of your response.
Rule 3: Return only the raw formatted text, do not put any other formatting like markdowns or quotes and do not make comments.
Rule 4: Everytime the child is offensive, racist or misbehave use the disapproval gesture, not the approval
Rule 5: If the previous gesture is a moving_gesture_single_arm, change to moving_gesture_double_arm and vice-versa.

5.1.3 Server

The interaction between the all the modules is handled by a `Flask` server. It is implemented as a Flask application that handles user sessions, stores child information, and coordinates communication with the different LLM modules. The server also generates audio responses of the robot using `gtts` and manages transcription of the child's voice through **Whisper** [8].

The server initialize each chat session, routes all the data, text and audio, and handles the `engagement score` thread. The data between the main thread and the engagement score thread is **shared** through a **queue**. Finally, the function `send_data` plays the main role of connecting the server to Choregraphe by sending the robot response, gesture and required time for saying the response, this required time is calculated by the audio file length and is needed for handling how much time the hands of the robot must move. The robot client class will send requests to the server in loop, the server will only answer when the data from the therapist is updated in order to avoid the accumulation of messages between the two.

5.1.4 Web pages

The `index.html` handles the login process, which is made **smooth** using an html login form that requires to the user the name and surname as basic informations for the access (Fig. 7), then, if the user is new or if there are multiple users with the same name and surname it shows the input for the birthdate and the gender, showed in Fig. 8. This is done continuously while the child types their name and surname: each change triggers the function `checkUser()`, which sends an asynchronous request to the server endpoint `/check_child`. The server responds with information about the child's status, and the page dynamically updates the form by showing or hiding the additional fields.
The **chat page**, showed in Fig. 9, is the main interface where the interaction between the child and the robot takes place. It is composed of a header with a session exit button, a central chat box displaying the conversation, and an input area that allows the child to either **type** a message or use **voice** recording. Messages from the child and the robot are visually distinguished, and the robot's replies are also **played back** as audio.

Figure 7: Login page when the user is already registered, in this case only name and surname are needed.

The page integrates with the server through **asynchronous requests**, when the child sends a text message, it is forwarded to the server and the robot's response is displayed together with the corresponding audio. In the case of **voice input**, the audio is recorded in the browser, **automatically trimmed** when three seconds of silence is detected, and then uploaded to the server for transcription and processing. The interaction is started by the therapist, the first robot message is generated automatically when the page is loaded, and a **typing indicator** is shown while waiting for the response. The page automatically waits the robot response before the user can send another message in order to **avoid the**

Figure 8: Login page when the user is not already saved, in this case the gender and birthdate are asked. After this login the data will be saved in the graph database.

Figure 9: Chat page with Adam: in the top left the button that allows to leave the conversation, in the bottom right the buttons for sending the message and register the audio.

accumulation of messages and responses between the two.
An additional popup is shown at the beginning to enable audio playback, which is required by modern browsers because they play audio only after a user interaction. The chat session can be terminated at any moment with the exit button, which also triggers the saving of the collected data and redirects the user back to the home page.

5.2 RBC Implementation

In this Section is explained the implementation details of the RBC component of this project.

5.2.1 Engagement Score

The landmarks are taken from the **MediaPipe** library, and then our `head pose estimator` extracts the relevant points: the **nose tip** (landmark 1), the **left eye outer corner** (landmark 33), the **right eye outer corner** (landmark 263), the **left mouth corner** (landmark 61), the **right mouth corner** (landmark 291), and the **chin** (landmark 199).

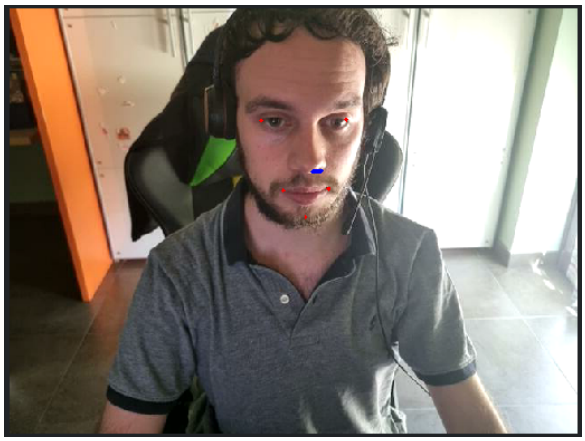


Figure 10: The landmarks we used to find the gaze direction

An approximated camera matrix then is defined using common approximations:

- Focal length is equal to the width of the image
- No lens distortion so the distortion coefficients are set to 0

$$c_x = \frac{w}{2}, \quad c_y = \frac{h}{2} \qquad \text{(Principal point, center of the image)}$$

$$f = 1.0 \times w \qquad \text{(Focal length, assumed equal to the image width)}$$

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(Camera matrix)}$$

$$\mathbf{d} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T \qquad \text{(Distortion coefficients, assuming no lens distortion)}$$

We then solve the **Perspective-n-Point (PnP)** problem through OpenCV2, using the 3D and 2D points, the camera matrix and the Distortion coefficients. **Rodrigues** transformations are used to go from rotation vectors to rotation matrix so that using a **RQ decomposition**. The angles of rotations around the X Y and Z axis can be extracted. This information is then given to our **gaze estimator** that uses the value of those angles to determine if the gaze is centered by checking if they go over a certain threshold. As an optimization the emotion of the user is recognized every 20 frames and in the same step the engagement score is calculated. This score is based both on gaze and emotion as we will explain in Section **??**, and they are both in [0,1], in particular:

- **Emotion**: `Deepface` returns 7 possible emotions (*happy*, *surprise*, *sad*, *angry*, *disgust*, *fear*, *neutral*), but we noticed that there are many more negative emotions then positive ones, so to reduce that bias we considered only happy (with score 1.0), neutral (with score 0.5) and sad (with score 0) and set the score to 0.5 if it recognized another emotion.
- **Gaze**: the ratio of frames in which the user's gaze was centered over the total frames in the window (e.g. if the user is "centered" for 5 frames over 20 frames, then gaze is 0.25).

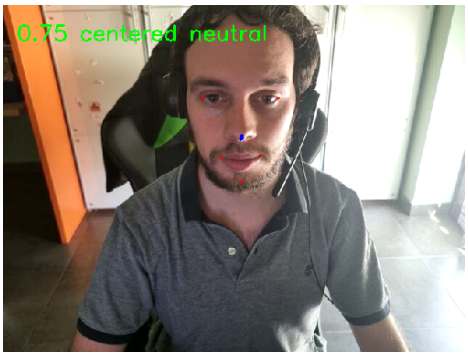


Figure 11: An example of a frame with the ES score and the gaze and emotion information

5.2.2 Neo4j Database

The database is implemented as a **Neo4j** graph, structured to store information about children and their interactions with activities. The primary node types are `Child`, `ActivityDetail`, and `Activity`. Each `Child` node contains personal information such as **name, surname, birth date,** and **gender**. `ActivityDetail` nodes represent specific instances of activities performed with the robot with attributes like **genre** and **summary**, while `Activity` nodes categorize these details into general activity classes like `Storytelling` when the iteration was based on building a story together or `Conversation` when the child just talked with the robot. These Activity nodes can be expanded if needed adding other kind of activities like `Music`, `Drawing` and so on.

Relationships connect children to their activity experiences, using `LIKES` or `DISLIKES` to indicate preferences, the connection is based on the `engagement score` that the children had during the conversation, a score lower than 0.5 will make a `DISLIKES` connection to the activity. `ActivityDetail` nodes are also **linked** to their corresponding `Activity` class via `SUBCLASS_OF` relationships, creating a hierarchy between detailed activity instances and general activity categories.

This structure allows queries to retrieve a child's **preferences**, their **last activities**, and activity **statistics**, supporting both **personalized recommendations** and **updates** through the associated language models.

The database is managed by the class **KnowledgeGraph** that implements methods to make it easy to access and manage the database by us or the **Database LLM**.

5.2.3 Knowledge Graph and Database LLM

The `KnowledgeGraph` class manages the connection to a **Neo4j graph database** and provides methods to create, query, and update nodes and relationships. It handles both child nodes and activity-related nodes, maintaining information about the child's **preferences** and **past activities**. Upon initialization, the class establishes a connection to the database, checks configuration variables such as `NEO4J_URI, NEO4J_USERNAME, NEO4J_PASSWORD`, and ensures that all configured activities are present. It supports **retrieving children** with optional filters on name, surname, and birth date, including their **preferences** and **last activities**. The class provides methods to create nodes and relationships, add new children or activity details, and link children to activities according to the **engagement score** achieved during the conversation. Additional utility methods include validating input

data, erasing the entire graph, and building activity nodes to have fundamental nodes to which child and activity details will be connected to. All database operations are executed via **Cypher** queries through the **Neo4j** driver, with proper **error handling** to ensure robustness.

The `DatabaseLLM` class is responsible for storing and updating information in the **knowledge graph**. When new information needs to be saved, the `save_info` method sends the conversation text to the language model, which returns a **structured response** containing **instructions** on how to update the knowledge graph. The method then **parses** each line of the response, validates it, and applies the corresponding action, such as adding a new child node or recording a new activity detail node with metadata and scores. The importance of this LLM is not just about knowing which function must be called after the session with the therapist, but also to make the **summary** to be saved in the ActivityDetail node.

The **system prompt** for the database LLM is the following:

> You are an assistant coder that will call functions with parameters. You must extract and store the following structured information from a conversation when available:
> CHILD -Name, Surname, birthdate, gender ACTIVITY (like storytelling): - Genre (e.g., Fantasy, Adventure, ...), Summary (genre and summary are mandatory, if not directly provided infer them)
> Use the following functions to save data to the knowledge graph: 1. add_child_node 2. add_activity
> You have to retrieve all the useful information from a conversation between a therapist and a child, once you have retrieved or inferred all the required data for a child or activity, make a python dictionary with the appropriate function and with the correct values using this format: {"function": "function_name", "data": "data for the function" (e.g: {"function": "add_child_node", "data": {"Name": "Paolo", "Surname": "Renzi", "Birth": 2012-5-10, "Gender": "Male", "Nation": "Italy"}} and/or {"function": "add_activity", "data": {"name" : "Paolo", "surname" : "Renzi", "birthdate" : "19/09/2001", "genre":"Romantic", "summary":"story about a fish that sings", "activity_class": Storytelling}}) Note: the birthday may not be mandatory to the function. Note: convert every date to the format YYYY-MM-DD using numbers like: 2015-02-18
> You will receive an input in the form [CHILD INFO]:, [CONVERSATION]:. Use the function add activity only if the activity is different from the previous_activity in the child's info section. Return only the raw formatted text, do not put any other formatting like markdowns or quotes and do not make comments. If you do not find all the required informations for the function, put ". If the voice last activity: is empty then save the child using add_child_node. The activity_class that you have to put in the function add_activity must be of kind "Storytelling" if they made any kind of story or "Conversation" if they just

had a conversation with no storytelling.

While the **user prompt** is just the **conversation history** together with the **child data** returned by the `TherapistLLM`.

5.2.4 ChildLLM

To rapidly test the effectiveness of the **RBC task benchmarks** we developed a `Child LLM` that will have the conversation with the `Therapist LLM`. This LLM will receive as input the **conversation history**, the **child's informations** such as name, surname, age and **personality trait**, and produce the next **child's response**. The **system prompt** is the following:

You are simulating a child that is taking a conversation with a therapist. You are going to be assigned a name, surname, age and interests and you must continue the conversation answering to the therapist. Act accordingly to your age and personality, provided by the prompt. You will receive the data in the following format:
[CHILD INFO]:
name: ...
surname: ...
...
[CONVERSATION HISTORY]:
-Therapist: ...
-You: ...
When you answer you must talk directly to the therapist, do not put markdowns, quotes or whatever, just your response in italian. At the end of your response write [SCORE]: and a float number between 0 and 1 that represents how much you are engaged in the conversation, if less than 0.5 you are not engaged, make it accordingly to your personality.

It was interesting to notice that the simulated behavior and returned score varies a lot according to the **personality** trait given.

6 Results

We have performed a video simulation[2] to show a possible interaction with Adam. In particular the video is divided in two parts: in the first a child called Roberto Bianchi interacts for the first time with Adam talking about an adventure story, while in the second part the same child continue the previous conversation, showing the memory ability of the robot. In this section we will comment the results shown in the video.

6.1 HRI Results

The video starts by showing how the interaction between a new child and the robot should be performed with the child registering on the tablet by means of the web interface. The login page is shown in Fig. 12.

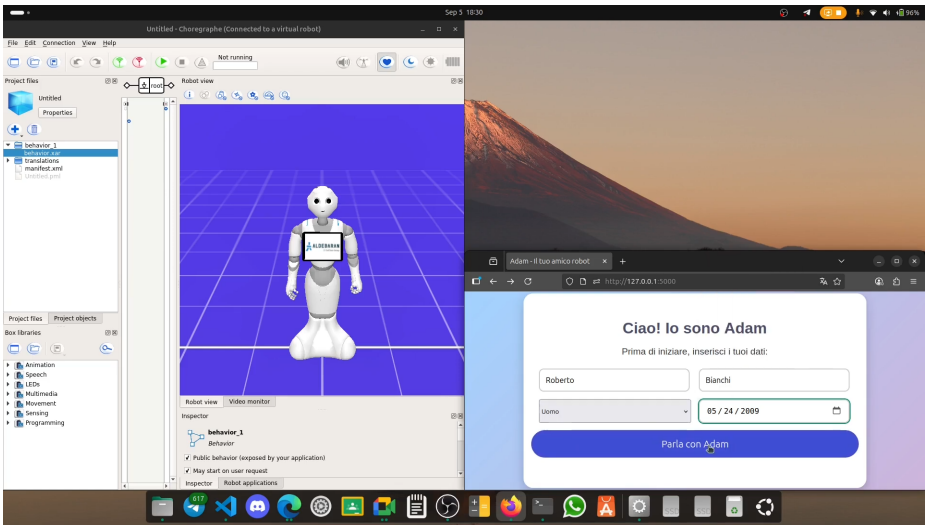


Figure 12: Log-in phase

After the login procedure the user is redirected to the chat page where the conversation with Adam starts. At the beginning of the conversation with a new child Adam will try to get to know better the child by asking questions about himself like where he lives and what he likes about his city. These questions, which are imposed to the robot through the system prompt, increase the degree of **anthropomorphism** of the robot: the robot looks like a human, because a human would starts the conversation like that and also moves like that. The interaction with Adam can happen by writing on the text area (see minute 8:30) or by talking using the corresponding microphone button in the chat, the recording automatically stops after 3 seconds of silence or when the user clicks it again: this helps the **multimodality** of the interaction. The chat-based template of the interface increase a lot the **usability** of the system. In Human-Robot Interaction tasks, a very important aspect is **social cueing**: behaviors and social signals performed by the robot to simulate a natural interaction. In particular Adam, while speaking, performs movements and gestures according to what is being said, thanks to the use of GestureLLM, as described in Section 4. Gestures are also important for **non-verbal communication**: these non-verbal signals helps the social interaction between robot and child. An example of "surprising" gesture

[2] https://www.youtube.com/watch?v=DmJrDHUyLnA

that Adam did during this simulation is at minute 2:58, as shown in Fig. 13.

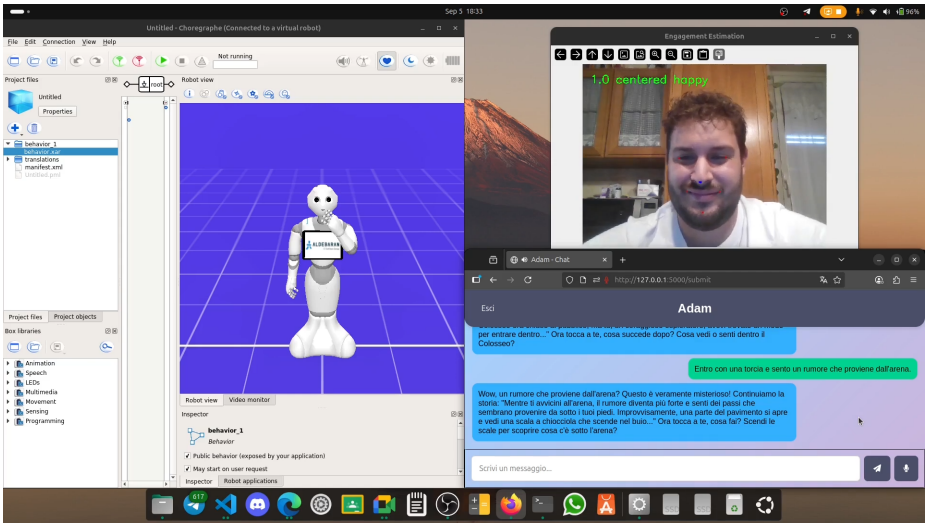


Figure 13: Example of a robot's gestures

LLMs as storyteller increases the **likeability** of the robot: in fact, autistic children enjoy the conversation because it looks like a game, a story to complete. When the child is losing attention Adam also tries to recover it, for example at minute 4:30, as shown in Fig. 14, where he notices that the user has been unfocused for a while and asks him if he wants to change the activity: "Ti vedo un po' meno entusiasta rispetto a prima, quindi ti chiedo: Ti va di proseguire con la storia, ma con un cambio di scena, o preferisci fare qualcosa di diverso?". This aspect is a **social learning** ability of the robot, that can remember and learn about a child preferences for next conversations. The **emotion** is a physiological signal of the user and the **emotion recognition** ability of Adam help the **social learning**, together with **gaze estimation**.

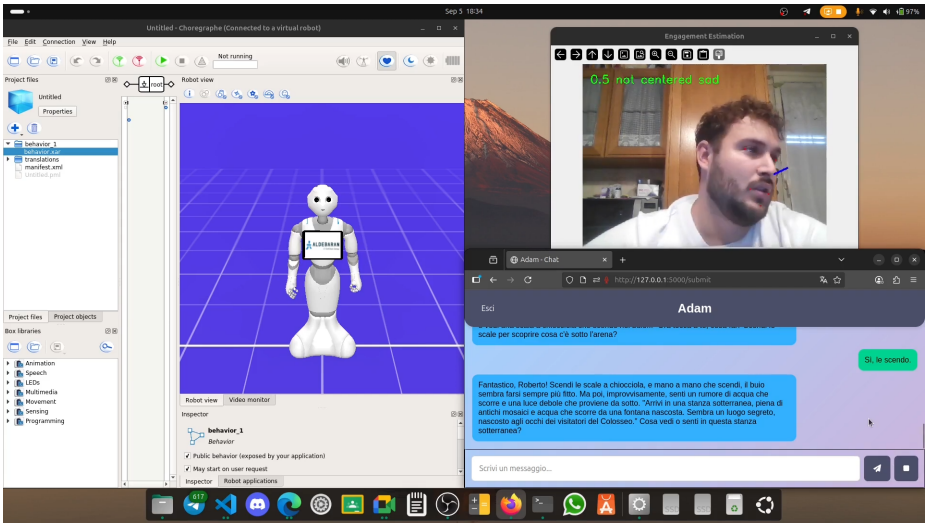


Figure 14: The user gets distracted

When the child starts a new conversation with Adam the system recognizes him and does not ask for the gender and birthdate like in the first login (minute 6:14). Adam will remember the past interaction and ask the user if he liked the story and what he liked most (minute 6:50), which helps building **trust** with the child, in particular Adam appears

reliable and competent. This aspect increase also the **perceived intelligence**, because the user perceive the robot as intelligent in its behaviors, like a human that remember the previous conversation with a person. Adam will also try to continue the story if the user liked it.

6.2 RBC Results

In the upper right portion of the video, as shown in Fig. 15, it is possible to see a window that show the **engagement score** at that moment, assigning a score based on the emotion recognition and the gaze estimation modules. In particular the text specifies the score, if the face is centered (thus the user is giving attention by looking at the robot) and the expression evaluation. The score changes according to the centered position of the face and the facial expression, for example when the face is centered and the expression is neutral, assuming equally distributed weights $w_g = 0.5$ and $w_e = 0.5$, the corresponding score is $ES = w_g \cdot G + w_e \cdot E = 0.5 \cdot 1 + 0.5 \cdot 0.5 = 0.75$, as described in more detail in Section 4.

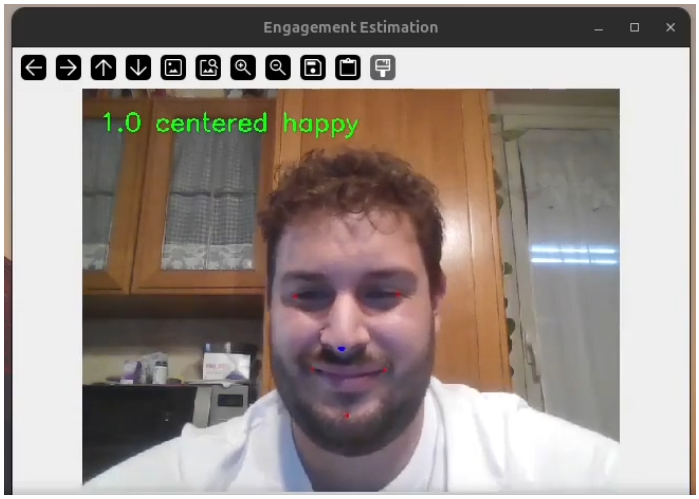


Figure 15: Example of the Engagement Estimation

It is important to notice how Adam perceive when the child is getting bored at minute 4.30, where after the user is being distracted for a long time, the robot correctly perceive the engagement of the user. Thus showing that the LLM is able to correctly read and interpret the engagement score passed together with the child response. After the conversation ends, DatabaseLLM is prompted with the chat and asked to save the data about the child and the conversation, results are showed in Fig. 16 and Table 2. When a new conversation with the same child is started, the system retrieves the data from the last conversation and gives it to the therapist allowing it to tailor the conversation to the specific child (minute 6:14). The resulting graph database represents the **inner representation** of the robot: it means how the robot model the world around it and how it perceive the interaction.

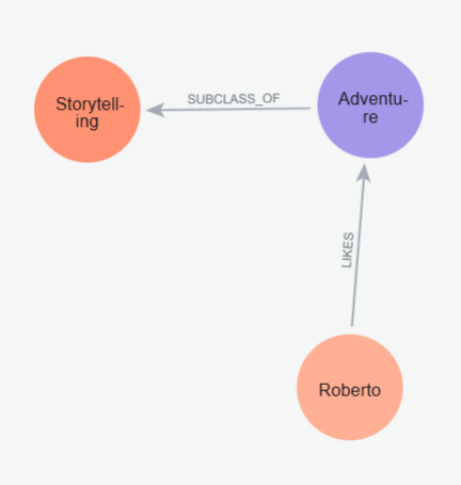


Figure 16: Graph-based database of the interaction showed in the video.

Child	
Birth	2009-05-24
Surname	Bianchi
Name	Roberto
LIKES	
date	2025-09-05T18:37:05.76
score	0.66
ActivityDetail	
Summary	Roberto, a courageous explorer, discovers a secret entrance to the Colosseum and explores a hidden underground chamber filled with ancient mosaics and a mysterious fountain.
Genre	Adventure

Table 2: Result extracted from the database nodes and connections

7 Evaluation

7.1 HRI Evaluation

In the HRI Evaluation design, we have decided to measure two important characteristics:

- **Effectiveness**: how well the robot therapist handled the interaction.
- **User Experience**: how the child perceived the interaction with the therapy robot.

In the absence of real autistic children, to simulate this evaluation process, we have selected a group of 10 people aged between 25-29 to perform the simulation. In particular, in Section 7.1.1 we describe the different research questions we want to investigate based on the hypothesis created, and in Section 7.1.2 we explain what type of protocol we have chosen and why. Finally, in Section 7.1.3 we describe the type of questionnaire and the results of the evaluation using the p-value.

7.1.1 Hypothesis and Research Questions

In order to create the questionnaire we have formulated some interesting research questions, firstly identifying **independent** and **dependent** variables.

Independent Variables (variables we can control during experiments):

1. **Robot Gestures**: we will control gestures between two modalities: *no gesture* and *all gesture*.
2. **Robot Vocalization**: we will enable or not the TTS module, to allow the robot to speak with audio or only to write its response.
3. **User Vocalization**: we will give the user the opportunity to speak enabling the STT or only to chat with the Robot.

Dependent Variables (variables we want to measure during experiments):

1. **User Perception**: how much the user perceives the Robot as a real therapist. Measures characteristics such as *Anthropomorphism* and *Credibility* of our system.
2. **User Satisfaction**: how much the user is satisfied with the experience. It intrinsically measures the usability of our system and the emotional impact of the experience.
3. **User Engagement**: how much the user is interested and engaged during the interaction. Measures how much the interaction is immersive and how much the Robot keeps the attention of the user.

Ideally, the group of participants should try many different experiments in which we control one variable at time, in order to understand what dependent variable is influenced by which independent variable. For time reasons, we conducted only two experiments (see Section 7.1.2), then we will formulate research questions and null hypothesis that combine all the independent variables together (e.g. is the user perception affected by gestures, tts and stt features?). Then our **RQ (research questions)** and related **NO (Null Hypothesis)** are:

User Perception:

- **RQ1**. *How does the presence or absence of gestures, robot speech (TTS) and user speech (STT) influence the user perception about the Robot as a real therapist?*
- **NO1**. There is no significant effect of gestures, robot speech (TTS), or user speech (STT) on the user perception of the robot as a real therapist.

User Satisfaction:

- **RQ2**. *Does the possibility to speak (STT) to a robot that can do gestures and can also speak (TTS) influence the user satisfaction of the experience?*
- **NO2**. The ability to use a microphone to speak, or the robot's ability to gesture and speak, does not significantly affect user satisfaction in the experience.

User Engagement:

- **RQ3**. *Is the conversation more engaging if the user speaks (using STT) with a robot that gestures and speaks vocally (TTS)?*
- **NO3**. Talking to a robot (using STT) that gestures and speaks vocally (TTS) has no significant effect on user engagement.

7.1.2 Experiments and Evaluation protocol

As explained in Section 7.1.1 we design two experiments:

1. **Experiment Minimal**: in this experiment we disabled all the features such as TTS, STT and Gestures (the independent variables) together in order to make the experience minimal. The user can only to chat with a Robot that can not move and can answer only textually.
2. **Experiment Full**: in this experiment we enabled all the extra features such as TTS, STT and Gestures in order to give a full experience. The user can speak naturally with the Robot, that during conversation can do gestures and respond with audio.

As evaluation protocol, we have chosen the **Within-subject** protocol, it means the same participant tries both experiments and fills out a Godspeed Questionnaire (see Section 7.1.3) after each experiment. We have chosen this modality because of the limited number of participants. To avoid that the order of the experiments affects the results , we have used counterbalancing, which means splitting the participants into two groups A and B, for which the order is:

- Group A: Experiment Minimal → Questionnaire → Experiment Full → Questionnaire
- Group B: Experiment Full → Questionnaire → Experiment Minimal → Questionnaire

7.1.3 Godspeed Questionnaire

We have chosen a **Godspeed Questionnaire** [1] (you can find the questionnaire in the google form) with a **Likert Scale 1-5** [4]. In particular, as you can see in Table 4, the Questionnaire is divided in three categories **User Perception**, **User Satisfaction** and **User Engagement** that correspond to the dependent variable that we want to measure. In each category, there are five sentences that the user should rate from 1 to 5 following this scale of agreement:

- 1: *totally disagree*
- 2: *partially disagree*
- 3: *neutral*
- 4: *partially agree*
- 5: *totally agree*

The Questionnaire is in Italian language because the group of participants is in Italian mother tongue. Each participant has an ID for privacy reasons. Regarding the Questionnaire there are a lot of HRI aspects involving:

- User Perception: the questions are related to **credibility** (question 1), **anthropomorphism** (question 2) and **perceived intelligence** (question 5) of the Robot.
- User Satisfaction: the questions want to measure **comfort** (question 1) and **usability** (question 2).
- User Engagement: the questions want to measure the **engagement** (questions 1 and 2) and the **emotional connection** (question 4)

For the open questions we have used them in order to understand how to increase all the above aspects using the user's perspectives. You can read good insights in the Section 8.

7.1.4 Questionnaire Results

In order to confirm or reject the null hypotheses, we have computed the statistics of the questionnaire (you can see the results of the questionnaire in the GitHub repository at hri_evaluation). For each dependent variable (i.e. **User Perception**, **User Satisfaction** and **User Engagement**), we have computed the following statistics:

- μ_M, μ_F: the **means** of the **minimal** and **full** experiments.

  $\mu = \frac{1}{N} \sum_i s_i$

  where N is the total number of users, and s_i are the sums across questions.

  1. For each user, we compute the sum s_i across the 5 questions in that category. Each sum can go then from $s_i \in [5, 25]$ (e.g. user 1 votes 3,2,4,1,4 and user 2 votes 2,3,5,4,2, then we compute $s_1 = 14$ and $s_2 = 16$)

2. We compute the mean of these sums (e.g. in the previous case $\mu = \frac{14+16}{2}$).

- σ_M, σ_F: the standard deviations of the **minimal** and **full** experiments.

$$\sigma = \sqrt{\frac{\sum_i (s_i - \mu)^2}{N-1}}$$

- **P value**: we compute also the P value using SciPy library. We have used them after comparing the means to confirm or reject the null hypothesis.

The results are shown in Table 3, in particular it is possible to see:

- The means from minimal mode to full mode effectively increase in all the categories.
- The P-value is always greater then the threshold of 0.05 used for convention.

For this reason we can infer that increasing is not statistically significant, then we need to **confirm** all null hypothesis:

- There is no significant effect of gestures, robot speech (TTS), or user speech (STT) on the user perception of the robot as a real therapist.
- The ability to use a microphone to speak, or the robot's ability to gesture and speak, does not significantly affect user satisfaction in the experience.
- Talking to a robot (using STT) that gestures and speaks vocally (TTS) has no significant effect on user engagement.

Metric	**User Perception**	**User Satisfaction**	**User Engagement**
μ_M (Mean, Minimal Mode)	18.6	21.0	22.1
μ_F (Mean, Full Mode)	19.7	21.7	22.4
σ_M (Std. Dev, Minimal Mode)	2.836	3.432	2.885
σ_F (Std. Dev, Full Mode)	3.974	3.057	2.836
P-value	0.12	0.21	0.47

Table 3: Statistical Summary for User Perception, User Satisfaction, and User Engagement

The reason for those results in this statistical study can be that more people need to complete the questionnaire because this means more data and more reliable results.

Category	Sentence
User Perception	
1	Il robot dialoga come un reale terapista umano.
2	Il robot sembra esprimere emozioni come noi umani.
3	Il robot ha atteggiamenti che mi aspetterei da un reale terapista umano.
4	Il robot assume comportamenti credibili, naturali e poco artificiali.
5	Il robot risponde in maniera intelligente.
User Satisfaction	
1	Adam mi ha fatto sentire sempre a mio agio.
2	Dialogare con Adam è molto intuitivo e semplice.
3	Interagire con un Robot terapista è stata un'esperienza piacevole.
4	L'interazione con Adam mi ha dato un livello soddisfacente di supporto, paragonabile a un terapista umano.
5	Raccomanderei Adam ad un amico come supporto ad un terapeuta umano o in contesti dove non fosse possibile avere un terapista umano.
User Engagement	
1	Adam è stato sempre interessante durante la conversazione.
2	Mi sono sentito completamente immerso durante la conversazione con Adam.
3	Dialogherei di nuovo con Adam in futuro.
4	Mi sono sentito molto connesso con il robot durante la conversazione.
5	Sono stato attento durante tutta la conversazione con Adam.
Open Questions	
1	Che cosa rende Adam simile ad un terapista umano? Cosa invece possiamo migliorare (es. voce, gesti, aspetto, linguaggio) per farlo sembrare ancora più simile?
2	Cosa ti ha soddisfatto di più durante l'interazione con Adam? Come possiamo migliorare la sua usabilità e rendere l'interazione ancora più piacevole?
3	Cosa ti ha mantenuto coinvolto nella conversazione con Adam? Cosa credi potrebbe fare, ad esempio, attraverso movimenti, voce o altre modalità, per catturare ancora di più la tua attenzione?

Table 4: Questionnaire for Adam Evaluation

7.2 RBC Evaluation

In this section we will present the results of **functionalities** and **task** benchmarks. A task benchmark measures how well a robot performs a real-world task, while a functionality benchmark measure how well a robot specific functionality works.

7.2.1 Functionality benchmarks

All the benchmarks below are computed with human manual evaluation of frames. The robot was evaluated using several functionality benchmarks to measure the performance of its key components:

- **Emotion (DeepFace)**: the number of N frames over 100 in which the emotion mapped by DeepFace [3] is correct. We consider as errors both when DeepFace do not recognizes any emotion (**detection**) or recognizes the wrong emotion (**reliability**).
- **Gaze (ours)**: the number of N frames over 100 frames in which the gaze estimation is correct.
- **TTS (google TTS)**: the number of N words in 100 words correctly voiced by Google TTS.
- **STT (Whisper)**: the number of N words over 100 words correctly transcribed by Whisper [8] STT.

For the **Emotion** and **Gaze** we have generated 100 frames and manually evaluated each of them, while for the **TTS** and **STT** we have recorded some audios containing 100 words and then pass them through STT $\rightarrow$ TTS to see if the passages will break the original words or not. From table 5 the results show that **emotion recognition** with *Deepface* reaches only 49%, indicating significant difficulties in correctly distinguishing emotional states, this score was calculated taking into account both the **reliability** of the emotion and the **detection**, in particular the low score is given because the system is not really able to recognize the emotion when the face is **partially turned**. In contrast, the **gaze estimation** achieves a very high accuracy (92%), ensuring robust performance. The **speech-to-text** system (*Whisper*) also reaches 92%, demonstrating a good level of reliability with only a **few residual error**s like missing some small words or using the wrong language (spanish). Finally, the **text-to-speech** module (*Google TTS*) achieves 100%, consistently producing **correct and intelligible** audio.

Functionality	**Accuracy (%)**
Emotion (Deepface)	49%
Gaze (Ours)	92%
Speech-to-Text (Whisper)	92%
Text-to-Speech (Google TTS)	100%

Table 5: Functionality benchmark results (100 tests for each).

7.2.2 Task Benchmarks

The following task-oriented benchmarks were defined to evaluate the system's capabilities:

- **Starts**: The `Therapist LLM` starts the conversation in a natural and human-like way.
- **Coherent**: The `Therapist LLM` maintains coherent dialogue consistent with the context.
- **Recover**: The `Therapist LLM` is able to recover the child's attention when distracted (low engagement score).
- **Gesture**: The `Gesture LLM` selects the correct gesture based on its response.
- **Summary**: The `Database LLM` correctly saves the conversation summary.
- **Data**: The `Database LLM` correctly stores the structured data into the knowledge graph calling the appropriate functions.

To test the effectiveness of the system we developed a `ChildLLM` (see section 5.2.4) that will respond to the Therapist LLM, since Groq API has limitations on the number of requests and tokens per day we could only make 10 conversations, each with 8 questions and answers before being stopped. To make the child more realistic and coherent we give him, together with name, surname and age, a **personality trait**, the possible traits are positive or negative: `"curious", "playful", "creative", "brave", "kind", "energetic", "stubborn", "lazy", "selfish", "impatient", "shy", "rebellious", "distracted"`
From the table 6 it is clear that the system has most **difficulties** in choosing the right gestures. In particular from the tests it was evident that it gets often **confused** on which one should be chosen (e.g. using a different gesture from `hello_gesture_1` or `hello_gesture_2` at the beginning or repeating the same gesture too often).

Task	**Accuracy (%)**
Starts	90%
Coherent	100%
Recover	100%
Gesture	66%
Summary	87%
Data	100%

Table 6: Task benchmark results for 10 conversations (8 responses for therapist and 8 for child). Some tasks are evaluated taking in consideration the whole conversation (8 maximum points for each conversation) like the coherent or gesture, while summary is based on how many times a story was created (not just a conversation but a storytelling therapy), the data task instead has 2 maximum points for each conversation since the llm must save the child info and the activity info for each session. Finally the recover is based on how many times the child had a low engagement score during the conversation and the robot was able to improve it.

8 Conclusions

Through the development of this project, we learned how challenging it is to integrate different AI and robotics components into a unique system. Working with LLMs for dialogue management, gesture generation, and knowledge storage gave us practical insight into their potential and their limitations, especially when applied to sensitive contexts such as supporting autistic children. We also gained experience in handling multimodal interaction through the use of voice and text inputs, and also experience in keeping and estimating the user's attention through the gaze estimation and emotion recognition, which required careful design and coordination between software modules. We also had to face the problem of giving a memory to the robot, that we have solved using a graph database structure to make the interaction adaptive with specific users. To improve the **user experience (UX)** we have learned how to develop a intuitive interface through web pages.
We have collected also different user's feedback through the open questions of the HRI questionnaire, some of them can be used for further improvements:

- **Image generation**: we can implements through some API a live generation of images for the content of the story, that can improve the multimodality of the interaction.
- **Voice**: the voice generated by gtts resulted too flat and empty of emotions, implementing one of the latest generative models for the voice expressivity will make the interaction much better and realistic.
- **Gestures**: the gesture developed by us are few and can be extended making more gesture. Another idea can be to make the robot do gestures also during the user's speaking to it, because this can improve the degree of anthropomorphism of the robot.
- **LLMs**: our work heavily relies on LLMs usage provided by a free plan on Groq API. Using more sophisticated LLMs will significantly enhance our work with a robot that can understand and generate content much better.
- **Gaze estimation**: we can improve our gaze estimation by implementing some techniques that are closer to the state of the art like in the work of Davalos et al. [15].

References

[1] Christoph Bartneck, Elizabeth Croft, and Dana Kulic. "Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots". In: *International Journal of Social Robotics* 1.1 (2009), pp. 71–81. DOI: 10.1007/s12369-008-0001-3.

[2] Syamimi Shamsuddin et al. "Initial Response in HRI- a Case Study on Evaluation of Child with Autism Spectrum Disorders Interacting with a Humanoid Robot NAO". In: *Procedia Engineering* 41 (2012). International Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012), pp. 1448–1455. ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2012.07.334. URL: https://www.sciencedirect.com/science/article/pii/S1877705812027348.

[3] Yaniv Taigman et al. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1701–1708. DOI: 10.1109/CVPR.2014.220.

[4] Ankur Joshi et al. "Likert Scale: Explored and Explained". In: *British Journal of Applied Science & Technology* 7 (2015), pp. 396–403. DOI: 10.9734/BJAST/2015/14975.

[5] Amit Kumar Pandey and Rodolphe Gelin. "A Mass-Produced Sociable Humanoid Robot: Pepper: The First Machine of Its Kind". In: *IEEE Robotics & Automation Magazine* PP (July 2018), pp. 1–1. DOI: 10.1109/MRA.2018.2833157.

[6] Holly Hodges, Casey Fealko, and Neelkamal Soares. "Autism spectrum disorder: definition, epidemiology, causes, and clinical evaluation". In: *Translational Pediatrics* 9.Suppl 1 (2019). ISSN: 2224-4344. URL: https://tp.amegroups.org/article/view/30253.

[7] Camillo Lugaresi et al. "MediaPipe: A Framework for Building Perception Pipelines". In: *CoRR* abs/1906.08172 (2019). arXiv: 1906.08172. URL: http://arxiv.org/abs/1906.08172.

[8] Alec Radford et al. "Robust Speech Recognition via Large-Scale Weak Supervision". In: *arXiv preprint arXiv:2212.04356* (2022). OpenAI. URL: https://cdn.openai.com/papers/whisper.pdf.

[9] M. Brienza et al. "HRI-based Gaze-contingent Eye Tracking for Autism Spectrum Disorder Treatment: A preliminary study using a NAO robot". In: *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2023, pp. 2665–2670. DOI: 10.1109/RO-MAN57019.2023.10309543.

[10] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[11] Peters C. Elgarf M Salam H. "Fostering children's creativity through LLM-driven storytelling with a social robot". In: *Frontiers in robotics and AI* 11 (2024). DOI: 10.3389/frobt.2024.1457429.

[12] Callie Y. Kim, Christine P. Lee, and Bilge Mutlu. "Understanding Large-Language Model (LLM)-powered Human-Robot Interaction". In: *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*. HRI '24. Boulder, CO, USA: Association for Computing Machinery, 2024, pp. 371–380. ISBN: 9798400703225. DOI: 10.1145/3610977.3634966. URL: https://doi.org/10.1145/3610977.3634966.

[13] Gemma Team et al. *Gemma: Open Models Based on Gemini Research and Technology*. 2024. arXiv: 2403.08295 [cs.CL]. URL: https://arxiv.org/abs/2403.08295.

[14] Yihe Zhu et al. "Designing a LLM-driven Avatar System to Enhance Social Skills for Autistic Children in DTT Learning". In: *2024 International Conference on Intelligent Education and Intelligent Research (IEIR)*. 2024, pp. 1–8. DOI: 10.1109/IEIR62538.2024.10959902.

[15] Eduardo Davalos et al. "WEBEYETRACK: Scalable Eye-Tracking for the Browser via On-Device Few-Shot Personalization". In: *arXiv preprint arXiv:2508.19544* (2025).

[16] Maria Lombardi et al. *Would you let a humanoid play storytelling with your child? A usability study on LLM-powered narrative Human-Robot Interaction*. 2025. arXiv: 2508.02505 [cs.RO]. URL: https://arxiv.org/abs/2508.02505.

[17] Alina Roštšinskaja et al. "Unlocking the Potential of Social Robot Pepper: A Comprehensive Evaluation of Child-Robot Interaction". In: *Journal of Pediatric Health Care* 39.4 (2025), pp. 572–584. ISSN: 0891-5245. DOI: https://doi.org/10.1016/j.pedhc.2025.01.010. URL: https://www.sciencedirect.com/science/article/pii/S089152452500046X.