UCLouvain

epl

École polytechnique de Louvain

GPTAlly: A Safety-Oriented System for Human-Robot Collaboration based on Foundation Models

Author: **Brieuc Bastin**
Supervisors: **Benoit Macq, Renaud Ronsse, Gustavo Alfonso Garcia Ricardez, Lotfi El Hafi**
Reader: **Jorge Solis**
Academic year 2023–2024
Master [120] in Electro-mechanical Engineering

Acknowledgments

I would like to express my sincere gratitude to Assoc. Prof. Gustavo GARCIA and Assoc. Prof. Lotfi EL HAFI for their guidance, feedback, availability, and invaluable assistance throughout the entire process of this Master's thesis. Their efforts in organizing such an incredible internship are also deeply appreciated.

I am also grateful to Prof. Renaud RONSSE and Prof. Benoit MACQ for their insightful feedback, encouragement, and for enabling the collaboration between the two universities.

I would also like to thank my colleagues at Ritsumeikan University and UCLouvain, particularly Eden MARTIN, for their participation in many enlightening discussions and for the support they provided during the thesis.

Lastly, I would like to extend my heartfelt gratitude to my family and friends for their unconditional support and understanding throughout this academic journey.

This work would not have been possible without the collective contributions of these individuals, and for that, I am grateful.

Use of AI Tools

In the preparation of this thesis, AI tools were utilized in accordance with Ecole Polytechnique de Louvain's guidelines[1]. More specifically, GPT-3.5 and Grammarly were used to enhance sentence fluency, readability, and to check for spelling mistakes. Additionally, GPT-4 was integrated into the system developed and discussed in this thesis, serving as a critical component of the implemented solution.

[1]EPL's guidelines: `https://moodle.uclouvain.be/pluginfile.php/174170/mod_resource/content/11/MasterThesis_Oct2023.pdf`

Contents

Abstract

Society 5.0 marks a transformative era where the focus shifts from productivity to enhancing the quality of life in the workplace through advanced AI and robotics. However, today's robots fall short of understanding their surroundings as humans do, primarily being restricted to pre-programmed tasks or dependent on supervised learning models with annotated datasets. Another challenge is the lack of safety metrics that consider the subjective perception of safety of users. Ensuring safety is paramount for physical Human-Robot Interaction, but the robot must not only be safe but also perceived as safe. This thesis proposes GPTAlly, a safe and adaptable system for human-robot collaboration that leverages innovative tools such as Large Language Models (LLMs) and Visual Language Models (VLMs). As LLMs can be seen as encoded representations of human language at scale, we investigate how they may be applied to infer a human's subjective perception of safety in collaborative tasks. The system facilitates adaptive safety evaluations based on users' safety perceptions by introducing a scaling factor suggested by an LLM into a Safety Index algorithm. When faced with potentially hazardous situations, such as moving too close to a user, the robot stops to prevent unwanted collisions. A streamlined coding paradigm based on an LLM determines the next action to avoid remaining stuck for prolonged periods. The decision is made either autonomously or influenced by the user, who can specify their preferences for the robot's actions in different situations using natural language. The system then shapes robotic arm trajectories based on user preferences by suggesting withdrawal or new objective 3D poses, implementing the actions that the robot can choose from. Through a user study, we compare a scaling factor representing the perception of safety computed with GPT-4 to an estimation given by the participants. The behavior of the robot is also assessed through various experimental setups to analyze the changes in robot behavior with different perceptions of safety. The accuracy of the streamlined coding paradigm is evaluated through contextual experiments, varying the number of conditions processed and paraphrasing the conditions. The satisfaction with the trajectories shaped from 3D poses is assessed through another user study. The results show that LLMs can effectively integrate human safety perceptions into safety evaluations. The perception of safety estimated by GPT-4 closely resembles the answers provided by the users, and most participants are satisfied with the changes in the robot's behavior. However, the results also indicate that the balanced accuracy of the streamlined coding paradigm in a given context can be below 50%. Finally, users preferred trajectories shaped through withdrawal methods influenced by their natural language over those without any influence. Codebase available at: `https://axtiop.github.io/GPTAlly`

Chapter 1

Introduction

Our society is aiming for Society 5.0[1]. Unlike Society 4.0, where the main goal is to achieve the highest productivity possible, sometimes at the cost of human well-being, this new human-centric society emphasizes workplace quality of life. Instead of robots and humans working separately, they are increasingly collaborating to complete tasks with the assistance of AI [1]. Robots are envisioned to assist with both physically and cognitively demanding tasks, reducing strain and pressure on human workers, thus fostering a healthier and more comfortable work environment. Achieving such tasks is challenging and requires innovative tools such as Large Language Models (LLMs) and Visual Language Models (VLMs). These tools exhibit emergent properties (such as natural language understanding, context awareness, and zero-shot prompting [2]) crucial for ensuring safe and productive Human-Robot Interaction (HRI) [3]. Indeed, robots are becoming smarter and safer owing to deployment at a large scale of Transformers in foundation models [4, 5, 6], alongside innovations spanning from the development of advanced motion planning algorithms [7] to reliable safety index metrics [8].

However, one of the main challenges with current robots is that they lack a human-like understanding of their surroundings and are still largely pre-programmed for specific tasks or they are governed by supervised learning-based models with annotated datasets [3]. Hence, they have limited capabilities to operate and adapt to new contexts in unstructured, human-centered environments. Integrating these capabilities into robots necessitates addressing technical difficulties related to real-time processing, contextual understanding, and dynamic decision-making. To overcome this challenge, emergent properties of LLMs, such as zero-shot prompting and context awareness, must be leveraged while ensuring robustness and adaptability [9].

[1]As defined by the Japanese cabinet office: `https://www8.cao.go.jp/cstp/english/society5_0/`

Another challenge is the complexity of natural language understanding (NLU) [3]. Indeed, natural language is highly contextual, ambiguous, and often relies on implicit knowledge, making it challenging for robots to accurately interpret and respond to commands with conventional coding approaches. Ideally, the robot should be able to recognize and understand natural language commands in a given context and adapt its behavior according to the user's preferences. With the incorporation of foundation models to deal with the extensive amount of data that sensors, especially natural language, can capture, robots may emulate the human ability to understand what the robot should do in a situation for which it has not been trained. Integrating foundation models into robotic systems requires addressing scalability, efficiency, and the ability to generalize across diverse contexts and languages.

The main challenge addressed in this thesis is the lack of safety evaluations that consider users' subjective perceptions of safety. While recent safety evaluation algorithms ensure safety for physical Human-Robot Interaction (pHRI) [10], most do not consider the perception of safety from the users. In alignment with the objectives of Society 5.0, which prioritizes the well-being of humans in the workspace, it becomes essential for robots not only to be safe but also to be perceived as safe. This integration involves managing language data's subjective and diverse nature with real-time human feedback into the safety evaluation framework. As LLMs can be seen as encoded representations of human language and culture at scale, they may be applied to infer the human's subjective perception of safety in collaborative tasks, which may differ from absolute objective safety. This entails developing algorithms and frameworks that capture and interpret human feedback, emotions, and contextual cues to accurately assess perceived safety.

In response to these challenges, this work proposes GPTAlly, a safe and adaptable system for human-robot collaboration. The first goal, which combines the challenge of the complexity of NLU and the incorporation of safety perception into safety evaluation, is to adapt the robot's behavior based on the user's perception of safety through natural language. If the user feels safe around robots, the robot moves quicker and is less restricted in his movements, unlike when the user feels unsafe around robots where the robot moves slower and is more restricted. When the user is in a dangerous situation, the robot is stopped to ensure safety. However, to avoid remaining stuck for a prolonged period of time, the robot can either wait for the situation to become safer, withdraw to a safer location, or change its current goal position by switching objectives. This decision can be made either autonomously by the robot or triggered by the user, who can specify his preference for the robot's actions in different situations with natural language. Finally, the system suggests withdrawal and target positions for the end-effector based on the user's desires in response to the first challenge where most of the current work

suggests models that are specifically trained for narrow tasks [11].

The main contributions of this work are:

- Demonstrate how **LLMs can incorporate human perception of safety into safety evaluations** by combining an established safety index with LLMs' interpretation of natural language input. This method allows the safety evaluation to incorporate the user's feelings and preferences.
- Assess the viability of utilizing LLMs for decision-making in HRC by **providing a streamlined coding paradigm**. This approach leverages the emergent properties of LLMs by diminishing the reliance on explicit condition descriptions and by using natural language to describe them.
- Show how LLMs can be used to **shape robotic arm trajectories by suggesting 3D poses based on natural language input**. The method allows to dynamically alter trajectories and decision-making based on natural language and the current state of the robot by suggesting 3D poses for the robot to withdraw or get an object.

The remainder of the thesis is organized as follows: Chapter 2 provides an overview of related works, including state-of-the-art models and systems, prompt engineering techniques, and safety related to HRC. Chapter 3 outlines the proposed method, detailing the system architecture and the implementation of each component. Subsequently, in Chapter 4, the experiments designed to validate the hypotheses outlined above are described in detail. Chapter 5 examines the results obtained from these experiments which are then interpreted and discussed in Chapter 6. Finally, Chapter 7 concludes the thesis, summarizing key findings, discussing their implications, and suggesting potential avenues for further investigation.

Chapter 2

Related Works

This section reviews the pertinent literature and previous studies that provide a foundation for this thesis. Firstly, state-of-the-art models and systems essential for data processing are explored. This includes an analysis of foundational models renowned for their generalization capabilities, task-specific models, and their integration into larger systems. Following this, various approaches to prompt engineering are discussed, highlighting best practices in the field. The section concludes by exploring safety concerns associated with the use of foundational models in robotics, recent safety evaluation algorithms, and the perception of safety in HRC.

2.1 Models and Systems for Data Processing

Developing intelligent robots that can interact with humans is a significant challenge in robotics. For such interactions to be adaptive and intuitive, the robot must be able to estimate the speaker's intentions and understand the meaning of sentences he has never seen or been trained for. More specifically, the research field called Symbol Emergence in Robotics (SER) has emerged in the past decades [12]. The SER is a human symbol system with emergent properties organized through physical and semiotic interactions between cognitive agents. Unlike top-down, supervised learning methods that can only be applied to narrow tasks and require annotated datasets, SER focuses on bottom-up, unsupervised Bayesian approaches to gradually enable robots to learn from interaction with the environment and users through their sensors. However, the recent breakthroughs in Transformers [4] and their application at large scale with foundation models such as GPT-4 [5] or CLIP [6] are supplementing the research field of SER with their strong adaptability and generalization capabilities.

2.1.1 Foundation Models

A foundation model is an AI model trained on broad data that can be applied to a wide range of tasks. It is programmed to function with a general contextual understanding of patterns, structures, and representations. This baseline of knowledge can be further exploited, modified, or fine-tuned to perform domain-specific tasks for just about any industry[1]. The first recent breakthrough that mainly impacted the research community was the training of foundation models made possible by the introduction of Transformers [4] which allows to improve the prediction quality while being more parallelizable, thereby requiring significantly less time to train.

The dissemination of Transformers to society was significantly advanced with the introduction of GPT-3, developed by OpenAI [5]. GPT, a large multimodal model capable of processing images and text inputs while generating text outputs, outperforms other language models in various academic tasks, as demonstrated in recent studies [5]. In [5], OpenAI undergoes comparison across a set of conventional benchmarks in Natural Language Processing (NLP) for GPT-3.5 and GPT-4. They both show human-level results on exams such as the SAT (Scholastic Assessment Test in evidence-based reading and writing). GPT-3.5 results are usually less accurate than GPT-4 as shown in the bar exam, where GPT-4 scores are in the top 10% of test-takers in contrast to GPT-3.5, which is in the bottom 10%. Both GPT-3.5 and GPT-4 exhibit promising potential for diverse applications in robotics, including text interpretation, task planning, and even code generation, as showcased in several recent papers incorporating GPT into their models. Table 2.1 compares leading NLP models across diverse academic evaluations. GPT-4 demonstrates significant superiority over existing language models optimized for specific benchmarks. **In this thesis, GPT-4 is the designated NLP model, primarily used to fulfill the roles of task planning and safety evaluation.**

Furthermore, foundation models are also used for object detection, which plays a crucial role in robotic computer vision by enabling the detection and localization of objects within images. Also developed by OpenAI, Contrastive Language-Image Pre-training (CLIP) [6] model is a VLM that allows the prediction of words in captions associated with images. CLIP is especially powerful when combined with other models to suggest object localization with their captioning [22]. Instead of labeling images, the Fast Segment Anything Model (FastSAM) [22] is used to segment images. Segment Anything Model (SAM) [23] has the disadvantage of being too computationally demanding. FastSAM was designed to offer a real-time solution for the segment anything task. It allows to segment an image 50x faster than SAM. MobileSAM [24] was later developed to have a model faster than FastSAM (sometimes called FasterSAM). It reaches results similar to FastSAM [24]

[1]Definition taken from: https://www.redhat.com/en/topics/ai/what-are-foundation-models

	GPT-4	**SOTA**
	Few shot	Few shot and best benchmark-specific tuning
MMLU [13] Multiple-choice questions in 57 subjects (professional & academic)	**86.4%** 10-shot	75.2% 5-shot Flan-PaLM [14]
HellaSwag [15] Commonsense reasoning around everyday events	**95.3%** 5-shot	85.6% ALUM [16]
HumanEval [17] Python coding tasks	**67.0%** 0-shot	65.8% CodeT + GPT-3.5 [18]
DROP [19] Reading comprehension & arithmetic	80.9% 3-shot	**88.4%** QDGAT [20]
MRCR [21] Multiround Co-reference Resolution	$\approx$ 60% N/A	$\approx$ **80%** Gemini Pro 1.5 [21]

Table 2.1: Overview of the performance of existing NLP foundation models on academic benchmarks. Table inspired from [5, 21].

while being 5x faster than the concurrent FastSAM and 7x smaller, making it more suitable for robot applications. However, MobileSAM lacks support for text prompts as input to correlate corresponding pixels with text, accommodating only point prompts. Another model that locates objects inside images is DETector with Image Classes (Detic) [25]. Detic uses text input to assign pixels from images to labels and offers more efficient image processing than FastSAM. **In this thesis, Detic is selected due to its large vocabulary and the imperative for fast processing.**

2.1.2 Task-Specific Models

Unlike foundation models, some models are tailored for specific tasks such as speech recognition or human pose detection. For instance, GPT-4 cannot process vocal modalities for transforming oral commands into text, necessitating the use of specialized models [5]. Similarly, GPT-4 cannot accurately estimate human body poses from images. While efforts are underway to develop models capable of such broad generalizations, with Gemini-1.5 [21], for example, they have yet to achieve accuracy comparable to the models discussed in this section.

The conversion of speech to text is essential to transcribe voice input into text that LLMs can process. Also developed by OpenAI, Whisper [26] is a robust speech recognition model that detects voice activity and predicts spoken words. While alternatives like Wav2Vec2 [27] and Hubert [28] offer similar performances on clean audios, studies suggest that both models exhibit higher Word Error Rates (WER) when subjected to increased audio noise [26]. Moreover, Whisper distinguishes itself with translation capabilities from various languages to English, offering valuable utility for robotic applications. **In this thesis, Whisper is favored for its performance in noisy environments, a crucial factor considering the typical surroundings where robots operate.**

Finally, a wide range of models and methods exist to estimate the body pose of humans. The monocular camera is the most widely used sensor for both 2D and 3D human pose estimation [29]. Two popular 2D estimation models are BlazePose from the framework MadiaPipe [30] and OpenPose [31]. They offer a 2D reconstruction of human poses from a single-view monocular image. BlazePose can run considerably faster at the cost of assigning some objects in the scene as human body parts [32]. However, the Root Mean Square Error (RMSE) remains close to the RMSE of OpenPose but with a much higher computational speed, which is advantageous for real-time robotic applications. Estimating and reconstructing 3D human poses from a single view of monocular images is a challenging task characterized by difficulties such as depth ambiguities, occlusion (from the body itself or other objects), and a scarcity of training data. Monocular cameras are not the only sensors that can be used for HPE. Models using depth and point cloud sensors have gained more attention recently because of the sensor's low cost and increased utilization [33]. State-of-the-art point cloud feature extraction techniques for classification and segmentation tasks are PointNet [34] and PointNet++ [35]. An alternative is to combine the 2D estimation from the monocular image with the depth sensor to compute the depth of the points measured in the image and map them to create a 3D representation. Another common sensor is wearable Inertial Measurement Units (IMUs), which can track the orientation and acceleration of human body parts by recording motions without occlusion. It should be noted that drifting may occur over time when using IMUs. **In this thesis, a monocular**

camera with the assignment of the depth a posteriori is used to allow real-time application while maintaining a high accuracy.

2.1.3 Systems based on State-of-the-Art Models

Depending on the task the robots must complete, they must be able to achieve a combination of task-specific operations such as code generation, open vocabulary object detection, high-level task planning, or even low-level control. The wide range of tasks requires using different state-of-the-art models which led to the development of various systems as shown in Figure 2.1. Two systems of interest that inspired the proposed method detailed in Section 3 are LAnguage Trajectory TransformEr (LATTE) [11] and Autoregressive Task and Motion Planning (AutoTAMP) [36].

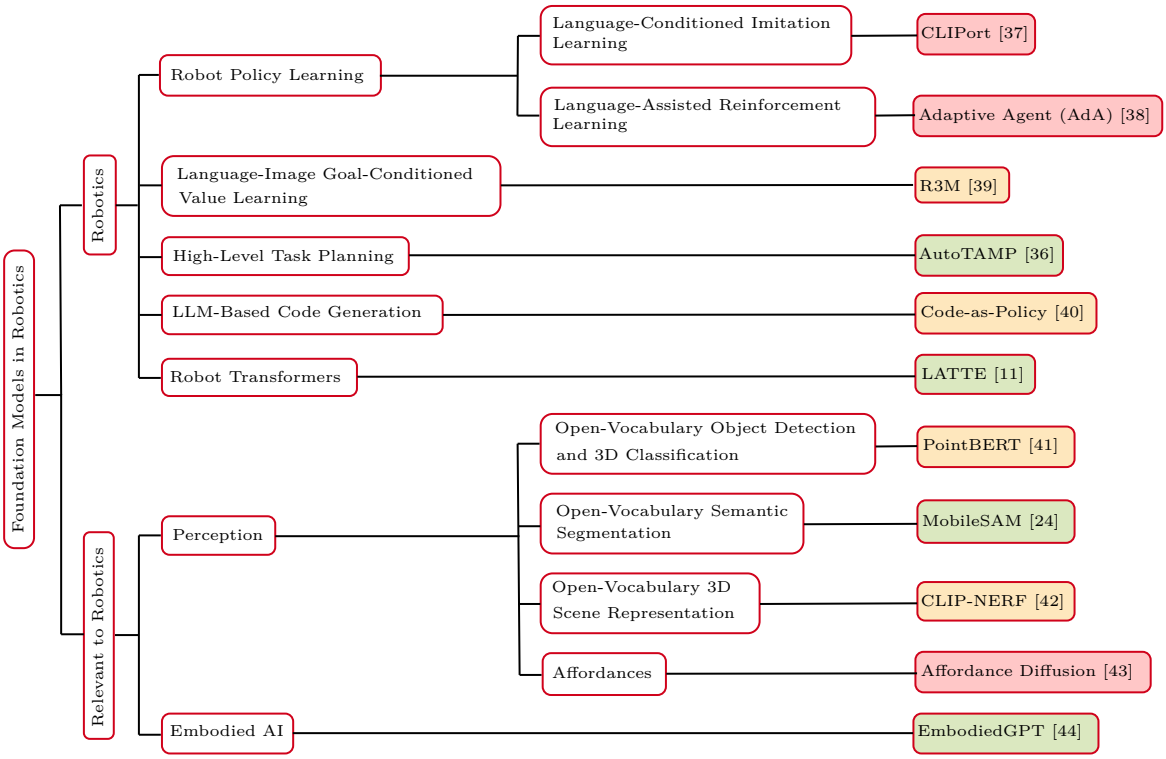


Figure 2.1: Overview of Robotics Tasks Leveraging foundation models. In green, the systems that are the most relevant to this thesis, in yellow, the slightly relevant systems, and in red, the systems that are not relevant to this thesis. Figure heavily inspired from [3].

LATTE [11] uses various VLMs and LLMs to modify a specific trajectory. Their general model utilizes pre-trained language models (BERT [45] and CLIP [6])

to encode the user's requests and target objects directly from a text input and scene images. Then, they concatenate the geometrical features generated by a transformer encoder network. Finally, they output new robot trajectories using a transformer decoder without requiring prior task-related information or information about the robot. Their model is trained on a specific dataset of trajectories and is thus prone to the challenge of Data Scarcity, especially since multiple modalities such as language, vision, and geometry are trained together. One assumption is made and tested for the model's training: a smaller but meaningful set of examples with semantically driven trajectory modifications can train an effective transformer decoder, given that the BERT and CLIP encoders have already been trained with large corpuses and can handle vocabulary and sentence variations. There are three types of oral commands which will be similar to the one used in this thesis:

1. Changes in the absolute Cartesian trajectory space: *e.g.*, "*go more to the right*", "*place it higher.*"
2. Changes in speed: *e.g.*, "*go faster*", "*go slower.*"
3. Positional changes relative to objects: *e.g.*, "*place it on my right*", "*place it closer to the red box.*"

Instead of training a new model from data to complete tasks, some systems use embodied AI such as Statler [46] or EmbodiedGPT [44]. **In this thesis, the system will not be fine-tuned on training data as one objective is to research the zero-shot capabilities of LLMs to suggest 3D poses.**

AutoTAMP introduces a framework leveraging LLMs to translate language task descriptions into formal task specifications through few-shot in-context learning. Additionally, it employs these LLMs as validators for identifying syntactic and semantic errors via corrective re-prompting. In robotics planning, tasks entail both high-level, discrete planning and low-level continuous motion planning. The concurrent resolution of these aspects is commonly known as task and motion planning (TAMP). The study concludes that LLMs are not optimal for directly resolving intricate TAMP challenges due to high planning time costs, particularly in scenarios necessitating multiple re-prompting iterations. **In this thesis, LLMs are utilized for single-sequence planning without corrective re-prompting, emphasizing real-time solutions rather than multi-task planning capabilities.**

2.2 Prompt Engineering

Prompt engineering has emerged as an indispensable technique for extending the capabilities of LLMS and VLMs. This approach leverages task-specific instructions,

known as prompts, to enhance model efficacy without modifying the core model parameters.

2.2.1 Types of Prompts

Manual and automated prompts are the two main categories of prompts that can be used with LLMs [2]. Manual prompts are typically authored by experts to furnish explicit instructions to the model, directing its focus on specific data types and guiding its approach to the task at hand. Automated prompts remove the need for human intervention through diverse algorithms and techniques, increasing the efficiency and adaptability of prompt engineering. In summary, manual prompts provide greater control over the output, while automated prompts are more efficient and adaptable.

Manual prompts prove highly effective when dealing with well-defined input data and when the output must adhere to specific structures or formats [2]. Nevertheless, manual prompts exhibit drawbacks, such as reliance on expertise and time, as well as susceptibility to significant changes in model predictions with minor prompt modifications [47, 48]. To overcome these limitations, researchers have devised various automated prompt-generation methods. Two prevalent types of automated prompts are discrete and continuous prompts. Discrete prompts rely on predefined categories to elicit responses, while continuous prompts factor in the ongoing conversation context to yield accurate outputs. **In this thesis, manual zero-shot prompting is employed to facilitate user interaction with the system and ensure greater control over the output.**

Zero-shot prompting offers a paradigm shift in leveraging LLMs [49] by removing the necessity for extensive training data, instead relying on meticulously crafted prompts to guide the model toward novel tasks [50]. In this approach, the model receives a task description within the prompt but lacks labeled data for training specific input-output mappings. Leveraging its pre-existing knowledge, the model generates predictions based on the provided prompt for the new task. For instance, the LAMA dataset introduced manually crafted close prompts to probe the knowledge of LLMs [51]. **In this thesis, zero-shot prompting is utilized to assess a safety factor based on human perception of safety, make informed robot decisions, and suggest 3D poses, all without additional training.**

2.2.2 Good Practices

In the context of prompt engineering, several studies suggest how to design effective prompts to provide the desired output [52, 53].

The following steps provide a structured and systematic approach to creating optimal and efficient prompts for LLMs:

- Defining a goal: a clearly articulated objective provides a guiding principle throughout the prompt engineering process.
- Understanding the model's capabilities: profound comprehension of the LLMs's limitations and capabilities, including the types of responses it generates (text, image,...), is paramount.
- Choosing the proper prompt format: clear, precise, and concise prompt formats are important, as well as equipping the model with essential details for coherent response generation [54].
- Providing Context: context furnishes the LLMs with enhanced comprehension metrics, facilitating prompt understanding and desired output generation.
- Testing and Refining: it is vital to rigorously test the prompt with the LLM used based on the defined goal [55].

In this thesis, the procedure suggested in [52] is followed to ensure efficient use of the LLMs.

2.3 Safety

In the rapidly evolving landscape of robotics and foundation models in AI, ensuring the safety of Human-Robot Interaction has become a paramount concern. As robots seamlessly integrate into various aspects of our daily lives (in our houses and at work), understanding, evaluating, and enhancing safety metrics have become imperative. Figure 2.2, which comes from [10], suggests a taxonomy of the currently existing methods providing safety in HRI. The safety considerations associated with the utilization of foundation models are considered. Then, the focus is shifted to the perception aspect with a human in the loop to adapt a safety metric. Finally, the nuanced psychological aspect of safety from the human perspective is briefly explored.

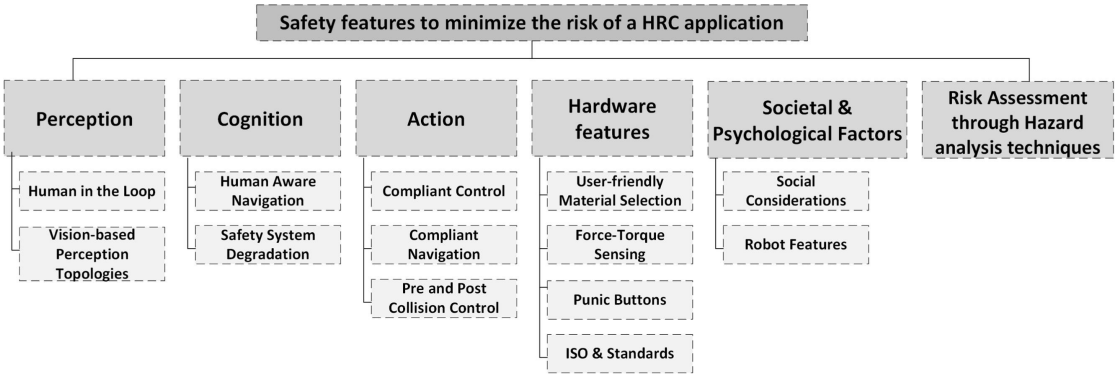


Figure 2.2: Taxonomy of the existing methods providing safety in HRI. Figure taken from [10].

2.3.1 Safety Challenges when Applying Foundation Models

Despite having strong generalization capabilities in vision and language processing that are useful to robotic applications, foundation models still face several challenges [3] which impact the safety of usage. There are 5 main challenges:

- Data Scarcity: it is challenging to obtain large-scale datasets for robot manipulation, decision-making, navigation, and other robotics tasks.
- High Variability: many different environments exist in which various robots operate. The foundation models must show generalization capabilities.
- Uncertainty Quantification: it happens for LLMs to hallucinate, which can lead to dangerous results in robotics. There is also a strong uncertainty regarding the output they provide.
- Rigorous Testing: foundation models must be rigorously tested before deployment, as the model is updated and as the robot operates in its target environments. It is however difficult (most of the time impossible) to test all the possible situations.
- Real-Time Performance: some foundation models have non-negligible inference time which could interfere with their deployment on robots.

The problem of safety evaluation in robotics is closely related to uncertainty quantification and hallucination. Caution is essential when utilizing language model results, especially in critical situations where human safety is at stake [5]. The reliability and safety of the system cannot be assured with outputs from GPT-4, as the model may yield varied responses for identical tasks, resulting in

unpredictability [56]. **In this thesis, the uncertainty in the responses is assessed and in the case of reasoning errors, there is a post-processing module in place.**

Before deployment, rigorous testing is crucial to ensure the safety of robotic systems interacting with humans. However, this task poses significant challenges for robots integrating foundation models, primarily due to two factors. These models can receive input across a vast spectrum, rendering it impractical to validate against all potential scenarios encountered during usage. Additionally, foundation models often exhibit unpredictable errors, complicating efforts to preemptively identify them. Hence, testing protocols must encompass a diverse array of scenarios to uncover potential flaws. Addressing the uncertainty inherent in foundation models, KnowNO [57] presents a framework designed to quantify and mitigate uncertainty in LLM-based planners. These planners can enhance their adaptability and reliability in real-world applications by acknowledging their limitations and requesting assistance when necessary. **In this thesis, thorough testing is conducted on the language by examining a wide range of natural language inputs, particularly through paraphrasing the prompts.**

After deployment, the safety can take the form of a failure prediction in a given scenario, which can allow the robot to deploy a safety-preserving fallback policy as proposed by [58]. **In this thesis, a constraint satisfaction module is implemented, forbidding the robot from making any dangerous movement due to dangerous answers from foundation models.**

2.3.2 Safety Evaluation

Ensuring safety during HRC is paramount, prompting researchers to develop diverse methods for assessing and controlling potential risks. Several safety evaluation methods and reactive strategies proposed in recent literature are highlighted with their respective objective related to the safety of human-robot collaboration.

There exist many different ways to estimate the danger in which the user currently is when collaborating with a robot. Based on the human pose estimation and the robot pose, [59] proposes a safety index to assess human safety by considering the consequences of the most dangerous situation. It links the potential danger to potential human injuries by assessing the severity of impact during a collision. Additionally, it foresees human movements to proactively limit the robot's speed in advance, thereby reducing the likelihood of a collision and minimizing the associated risks. Using the robot state, human state, and worst possible action is not the only method for determining the safety index. For example, an index based on impact force and impact stress was implemented by [60]. [61, 62, 63] estimate safety using injury evaluation, hazard analysis, or the distance and robot link momentum. **In this thesis, an innovative safety index is suggested by incorporating a**

scaling factor suggested by An LLM based on the user perception of safety into an SI algorithm suggested by [59].

Robots have different reactive strategies which can be driven by different factors. For example, a safety index, which considers inertia and the distance between the centers of mass of the human and robot was developed by [64]. This work inspired another work named Asymmetric Velocity Moderation (AVM) for human-safe robot control [65]. However, AVM uses multiple distances distributed along both human and robot bodies to calculate a specific restriction for every direction of the robot's motion. These work on the control of robots with a focus on safety are not the only ones. There exist also [66] who made the robot compliant to external forces where the safety of the interaction is based on reducing the impact severity when a collision occurs. Another example is [67] which proposes a time-optimal control policy based on Time Optimal Path Parameterization. **In this thesis, the reactive strategy involves adjusting speed based on a safety index, which is influenced by users' perceptions of safety through the use of LLMs.**

2.3.3 Perception of Safety

As mentioned in [10], human perception plays a vital role in the adoption and acceptance of collaborative robots. This significance becomes particularly pronounced when human-robot collaborative tasks unfold in demanding and dynamic environments, characterized by time-critical conditions, such as industrial production lines. Guaranteeing safety is only the first step toward seamless physical Human-Robot Interaction (pHRI). The next important aspect to consider is that a robot must be perceived as safe, in addition to actually being safe. Safety perception is defined in [68] as a synonym of Psychological safety: "*It ensures that the human perceives interaction with the robot as safe, and that interaction does not lead to any psychological discomfort or stress as a result of the robot's motion, appearance, embodiment, gaze, speech, posture, social conduct, or any other attribute.*"

To measure the level of perceived safety in a pHRI experiment, different methods have been used in the literature as explained in [68] and they are divided into four broad categories: questionnaires, physiological measurements, behavioral assessment, and direct input devices. In the realm of pHRI, a prevalent assessment approach involves utilizing questionnaires or surveys. In a typical HRI experiment, participants can be asked to fill out a pre-interaction questionnaire. These questionnaires may encompass inquiries about participants' demographic information, such as age, gender, and height as well as previous interactions with robots. Following their interaction with the robot, participants are requested to complete a post-trial questionnaire, aimed at capturing their reflections on the human-robot interaction (HRI) experience and their perceptions of the robot. The second category involves measurements of physiological signals such as the heart

rate, Galvanic skin response, eye gaze, muscle response, and respiratory rate. A typical measure used in behavioral assessment is the distance that the human keeps from the robot, which is intuitively inversely proportional to the sense of perceived safety [68]. The last category comprises devices designed to offer immediate feedback during the experiment. Similar to questionnaires, these devices offer a subjective gauge of perceived safety but differ in that they can provide this assessment in real-time, rather than waiting until the conclusion of the experiment. **In this thesis, pre-interaction and post-interaction questionnaires will be filled by the users based on videos.**

In the context of industrial manipulators, there is a general enhancement in the perception of safety when the relative distance between humans and robots is substantial, mostly when surpassing a specific threshold (*e.g.*, 2 m in [69]), and when the robot operates at low speeds, usually below a designated threshold as observed in [69]. Intuitively, human subjects tend to accept the robot's higher acceleration and speed when the relative distance is increased. The size of the robot also plays a role, with smaller robots being perceived as less threatening and allowing for higher speeds to be considered safe. A significant factor in perceived safety is the fluency and predictability of the robot's motion, which is further improved by providing some form of communication to the human. Furthermore, participants tend to feel safer when they have prior experience with the same task. **In this thesis, there will be participants with and without prior experience with robots.**

Chapter 3

Proposed Method

The overall goal of the proposed method is to offer a safe and adaptable system for human-robot collaboration. The approach is designed to be independent of any particular robotic arm. Based on a user's natural language command, the position of objects, the position of the user's limbs, the end-effector pose, and the ongoing action of the robot, the trajectory and safety evaluation must be modified accordingly.

3.1 Problem Definition

This section presents the different variables utilized to construct the functions described in Equation 3.1 that incorporate the system.

Let L_{in} be the user's natural language input sent to express the user's desires during the experiment, such as L_{in} = "*I am very afraid of robots, please be careful*" or L_{in} = "*When you withdraw, withdraw to the left.*" Let $GPT_f \in [0:2]$ be a scaling factor provided by GPT-4 with 0 meaning that the user is confident enough to remove the safety evaluation and 2 meaning that the user does not want the robot to move in the same workspace as the user. Let $P_{\text{human}} \in \mathbb{R}^3$ be the human limb position the closest to the robot, $P_{\text{robot}} \in \mathbb{R}^3$ be the position of the end-effector, and $P_{init} \in \mathbb{R}^3$ be the current objective position where the end-effector is moving. They are all expressed in the Cartesian frame positioned at the base of the robotic arm. Finally, $SI \in [0:1]$ is the safety index with 0 being the most dangerous and 1 being the safest.

Let $\mathbf{O} = \{O_1, ..., O_M\}$ be a collection of M objects in the environment, each with a corresponding position $P_{O_i} \in \mathbb{R}^3$. Each object is associated with a text caption provided by a text input from the user before the interaction begins. The poses are expressed in the Cartesian frame at the base of the robotic arm. Furthermore, $F_{\text{choice}} \in \{1, 2, ..., i\}$ is the index i of the function h_i to be executed by the robot.

It is determined by the decision-making module implemented with GPT-4.

Lastly, $P_{obj} \in \mathbb{R}^3$ is the suggested objective position where the end-effector should move. $P_{in} \in \mathbb{R}^3$ is the new objective position where the end-effector has to move after P_{obj} is processed by the constraint satisfaction module. $V_{in} \in [0:1]$ is a scaling factor computed by the constraint satisfaction module to ensure that the speed is scaled with the SI. The positions are represented in the Cartesian frame at the base of the robotic arm.

$$\begin{cases} SI = f(L_{in}, P_{init}, P_{\text{human}}, P_{\text{robot}}), \\ F_{choice} = g(L_{in}, P_{init}, P_{O_i}, P_{\text{human}}, P_{\text{robot}}), \\ P_{obj} = h(L_{in}, P_{init}, P_{O_i}, P_{\text{human}}, P_{\text{robot}}, F_{\text{choice}}). \end{cases} \tag{3.1}$$

The first goal is to design a function f that evaluates the safety (SI) based on the robot's position, the current objective position, the human limbs' position, and the oral commands. The second goal is to design a function g that maps the current state of the robot to a specific function F_{choice}. Lastly, the function h implements the functions with the index F_{choice} and suggests the position to which the robot should go.

3.2 Proposed System Architecture

The proposed system represented in Figure 3.1 uses the functions f, g, and h from Equation 3.1 to provide a safe and interactive system. These functions are non-trivial and uncertain as they integrate data from diverse modalities and the solution space has multiple solutions that fulfill the user's semantic preferences. The model architecture is divided into an input module (in yellow), three function modules (one per function), and a constraint satisfaction module:

- **Input module:** multimodal perception module that comprises a speech recognition model (Whisper [26]), which transforms natural language input into text, and two VLMs. The first VLM (Detic [70]) is used to estimate the position of the objects on the table whereas the second VLM (MediaPipe [71]) is used to detect the position of the human body parts.
- **Second module:** associated with the function f (in red), it is a safety evaluation module augmented with GPT-4 [5]. GPT-4 interprets the user's perception of safety by evaluating a scaling factor GPT_f used in the safety index (SI) algorithm.
- **Third module:** GPT-4 is used to interpret the current situation and the user's desires to decide on the next action (associated with the function g, in green).

- **Fourth module:** the robot computes the new objective position (P_{obj}) of the robot based on the user's instructions with the use of GPT-4 (associated with the function h, in black).
- **Constraint satisfaction module:** the suggested position (P_{obj}) enters a constraint satisfaction module that ensures the positions remain in the workspace and are not dangerous to the user. The maximum speed of the robotic arm requested must also respect the constraints provided by the user.

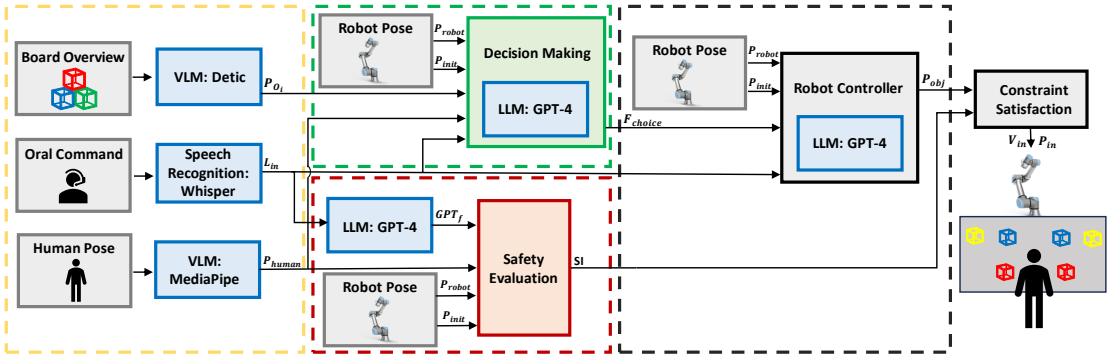


Figure 3.1: System architecture: in yellow, the input module is composed mostly of pre-trained models. In red, the safety evaluation module comprises an SI algorithm and a GPT factor (GPT_f) computed by GPT-4. In green, the decision-making module is designed with GPT-4. In black is the robot controller module, which implements the robot functions that enter the constraint satisfaction module.

3.3 Multimodal Perception

The multimodal perception functions of the proposed method involve the collection of two distinct types of data: proprioceptive and exteroceptive data. Proprioceptive data, inherent to the robot system, encompasses the positioning of each joint within the robot and the Cartesian coordinates of the robot's end-effector, denoted as P_{robot}. While fundamental to the experiment, further elucidation of this data is unnecessary due to its embedded nature within the system. In contrast, exteroceptive data is the focal point of this section. This data type comprises sensory inputs obtained from oral commands and visual inputs. Initially, user commands are captured by a microphone, denoted as L_{in}, which subsequently transmits them to an LLM (Whisper [26]) for audio-to-text encoding. Furthermore, a depth camera captures images of the board and the objects positioned on it, relaying them to a VLM (Detic [25]). Utilizing provided captions, Detic identifies and locates the objects on

the board. Finally, another camera, directed towards the user, detects human body parts, denoted as P_{human}. This task is accomplished by utilizing another VLM named BlazePose from MediaPipe [32].

3.3.1 Speech Recognition

Whisper is an audio-to-text model that predicts spoken words within a given audio snippet. In addition to its word prediction capabilities, Whisper also integrates voice activity detection, simplifying its application for this specific purpose. Notably, Whisper promptly initiates audio-to-text encoding whenever the user begins speaking, ensuring seamless transcription of spoken content. To manage resources efficiently, recordings are configured to cease after a maximum duration of 5 seconds, following which the audio is processed by GPT-4.

The processing by GPT-4 is done to evaluate for which module the audio snippet is useful. Indeed, the safety evaluation (red), decision making (green), and the robot controller module (black) use L_{in} as input to guide their answer. Still, they should not update their user input every time L_{in} is updated for another module. This is done with the help of GPT-4 with the prompt detailed in Appendix 8.1. GPT-4 takes the text as input and publishes the data on the corresponding topic. For example, "*I am very afraid*" is sent to the Safety Evaluation module, "*I want you to take a block that is close to me*" is sent to the module responsible for changing objectives (in the decision-making module and robot controller module).

3.3.2 Scene Overview

In this system architecture, Detic localizes objects within the image and subsequently assigns labels to them. To enhance prediction accuracy, users can input labels in text format at the onset of the interaction with the robot. Following successful localization and labeling of objects within the image, the depth information from the camera is leveraged to calculate each object's Cartesian coordinates. Detic operates on a single thread, ensuring the fastest processing possible, and publishes its estimations on a designated topic. These estimations then serve as inputs for the main loop, denoted as P_{O_i}.

3.3.3 Human Pose

The estimation of Cartesian coordinates for human body parts is facilitated through the utilization of BlazePose from MediaPipe [32]. This model is structured to identify and map 33 key points distributed throughout the human face, torso, and legs. However, only nine key points are employed to construct an avatar representing the user. An illustrative example of the model's configuration on

an image is depicted in Figure 3.2. Following the identification of key points on the image, the depth data captured by the camera is employed to compute the Cartesian coordinates corresponding to each body part.

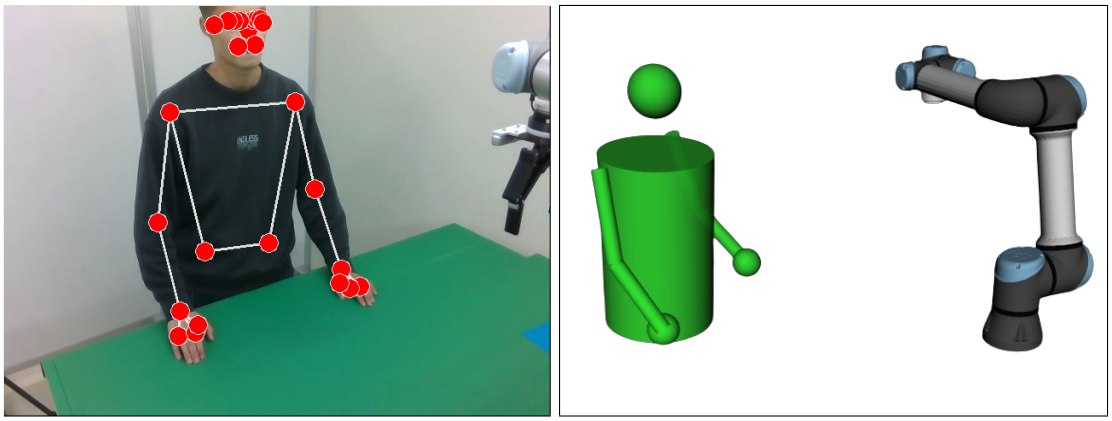


Figure 3.2: (Left) Example of a human pose estimation with MediaPipe [32]. (Right) Rviz representation of the avatar from the corresponding example: a human is represented with cylinders and spheres.

Utilizing the measurements obtained from the 9 key points, a representation of the human body is constructed in the form of an avatar using primitive shapes. This avatar is represented using spheres and cylinders, as illustrated in Figure 3.2. Specifically, the avatar comprises two spheres for each hand and one sphere for the head. The smaller cylinders represent the forearms and arms, whereas the larger cylinder depicts the user's body. To determine the center of the body cylinder, the intersection of lines connecting the shoulders and hips is computed. This approach proves efficient for computation while maintaining a reasonable level of accuracy in representing the user's body parts.

Finally, to prevent misidentification of the user's location, each body part must fall within a specified proximity to predetermined coordinates set at the beginning of the experiments. This ensures accurate placement of the avatar relative to the user's actual position. If the model locates the user too far from the predetermined coordinates, P_{human} is not updated and the previous localization is kept.

3.4 Safety Evaluation

In a first attempt to evaluate the user's safety, the safety index computation solely relied on GPT-4, utilizing the current state and user language input, but failed to display real-time capabilities. Indeed, the safety index must be very reactive as it is responsible for the user's safety. The limitation of GPT-4 being not real-time

as mentioned in Section 2 makes it a limitation for this application. Instead of computing the safety index with GPT-4 every iteration of the safety loop, a scaling factor, $GPT_f \in \mathbb{R} \rightarrow [0:2]$ is computed whenever the user provides a new input prompt related to safety.

The safety evaluation proposed is based on a Safety Index algorithm proposed by [59] with the incorporation of GPT_f which scales the danger score as shown in Equation 3.4. The prompt used to suggest the scaling factor based on the user voice input is detailed in Appendix 8.2. The original method associates the danger to potential human injuries by calculating the impact severity of a collision and by anticipating the human motions to restrict the robot's speed closer to impact, which decreases the risk of a collision. The output is associated with the impact severity of the worst human action. On the left of Figure 3.3, one can see the original algorithm without the GPT factor, and on the right, GPT_f is added as an input of the Danger Score (DS) function.

Algorithm 1 Safety Index, Original

Input: $x_{h0}, v_{h0}, x_{r0}, v_r^*$
Output: SI
$\hat{a}_h \leftarrow WHA(x_{h0}, v_{h0}, x_{r0}, v_r^*)$
$x_h' \leftarrow x_{h0} + v_{h0}\Delta t + \frac{1}{2}\hat{a}_h \Delta t^2$
$v_h' \leftarrow v_{h0} + \hat{a}_h \Delta t$
$x_r' \leftarrow x_{r0} + v_r^* \Delta t$
$SI \leftarrow -DS(x_h', v_h', x_r', v_r^*)$

Algorithm 2 Safety Index, Modified

Input: $x_{h0}, v_{h0}, x_{r0}, v_r^*, GPT_f$
Output: SI
$\hat{a}_h \leftarrow WHA(x_{h0}, v_{h0}, x_{r0}, v_r^*)$
$x_h' \leftarrow x_{h0} + v_{h0}\Delta t + \frac{1}{2}\hat{a}_h \Delta t^2$
$v_h' \leftarrow v_{h0} + \hat{a}_h \Delta t$
$x_r' \leftarrow x_{r0} + v_r^* \Delta t$
$SI \leftarrow \|DS(x_h', v_h', x_r', v_r^*, GPT_f) - 1\|$

Figure 3.3: (Left) Original algorithm proposed by [59]. (Right) Modified safety index algorithm with the incorporation of GPT_f in the DS equation.

where x_{ho} and v_{ho} are respectively the initial position and speed of the closest human body part to the robot, x_{ro} and v_r^* are the initial position and control input of the end-effector of the robot, and WHA is computed with Equation 3.2:

$$\begin{aligned} \underset{a_{h,t}}{\text{minimize}} \quad & F\left(x_h(a_{h,t}, t), x_r(t)\right), \\ \text{subject to} \quad & \|a_h\| \leq A_h, \ 0 < t < \Delta t, \end{aligned} \tag{3.2}$$

where $F(\mathbf{x}_{h0}, \mathbf{x}_{r0}) = \frac{1}{2}\|\mathbf{x}_{h0} - \mathbf{x}_{r0}\|^2$, A_h is the maximum human acceleration, and Δt is the short-term human behavior estimation time. Finally, the initial and modified danger scores are computed with Equation 3.3 and Equation 3.4 respectively.

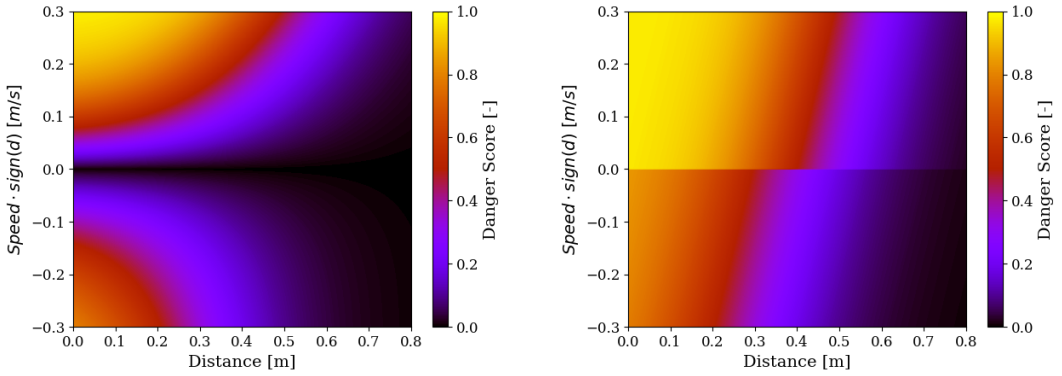
$$DS = \begin{cases} 2 & \|\mathbf{d}\| = 0, \\ f_S(\|\mathbf{J}\| \cdot f_G(\|\mathbf{d}\|)) & \|\mathbf{d}\| > 0, \frac{\pi}{2} < \phi \leq \pi, \\ f_S((1-\rho) \cdot \|\mathbf{J}\| \cdot f_G(\|\mathbf{d}\|)) & \|\mathbf{d}\| > 0, 0 \leq \phi \leq \frac{\pi}{2}, \end{cases} \tag{3.3}$$

where f_S and f_G are respectively the Sigmoid and Gaussian functions, $\mathbf{d}$ is the displacement vector between the end-effector and the closest human body part, $\rho \in \mathbb{R} \rightarrow [0:1]$ is a human trust factor, the norm of $\mathbf{J} = -m(\mathbf{v}_{h0} - \mathbf{v}_{r0})$ represents the numerical value of the impact severity, $\phi = \arccos\left(\frac{\mathbf{d} \cdot \mathbf{v}_{\text{rel}}}{\|\mathbf{d}\|\|\mathbf{v}_{\text{rel}}\|}\right)$, and $\mathbf{v}_{\text{rel}} = \mathbf{v}_h - \mathbf{v}_r$. The GPT factor ($GPT_f$), which is incorporated in this thesis, is computed with GPT-4 and is used to scale the DS. Equation 3.4 shows how DS is computed with GPT_f. In comparison to the SI provided by [59], this method incorporates four distinct modifications:

- **The SI is the absolute value of DS - 1**: this makes $SI = 0$ the most unsafe situation and $SI = 1$ the safest situation, always keeping SI positive.
- **Instead of multiplying f_G and $\|\mathbf{J}\|$, they are a weighted sum**: this allows a lower SI when the robot is close to the user even at lower speeds. This is achieved by setting a higher weight multiplying f_G than the weight multiplying $\|\mathbf{J}\|$. Changing the multiplication into a weighted sum also increases the SI when the robot moves away from the obstacle, as shown in Figure 3.4. This behavior is not the same when the functions are multiplied as the SI decreases when the speed increases, even when going away from the obstacle.
- **Inside of f_S, GPT_f multiplies the weighted sum**: this takes into account human desires and feelings. GPT_f is inside of f_S to ensure the SI (and DS) remains bounded between 0 and 1. Adding GPT_f and not simply modifying ρ offers the user more flexibility regarding the DS function's shape. Indeed, the SI is influenced by user preferences both when the robot is moving away from the user and when moving toward the user. The DS function can also be adapted for each user with GPT_f and for different applications with ρ.
- **Only consider the closest point on the human body, not the most dangerous one**: this enables a shorter computational time at the cost of degrading slightly the evaluation of safety [65]. If the nature of the task makes this modification too dangerous, it can be modified back to the original method.

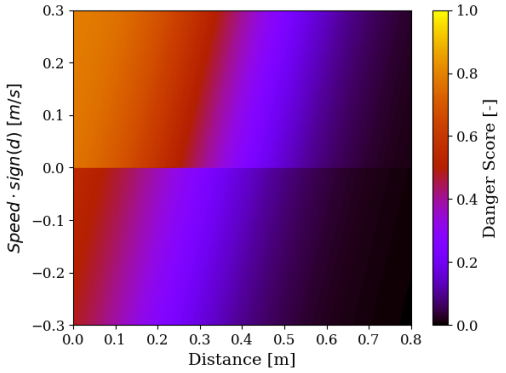
$$DS = \begin{cases} k_0 & \|\mathbf{d}\| = 0, \\ f_S((\|\mathbf{J}\| \cdot k_1 + f_G(\|\mathbf{d}\|) \cdot k_2) \cdot GPT_f) & \|\mathbf{d}\| > 0, \frac{\pi}{2} < \phi \leq \pi, \\ f_S((1-\rho) \cdot (\|\mathbf{J}\| \cdot k_1 + f_G(\|\mathbf{d}\|) \cdot k_2) \cdot GPT_f)) & \|\mathbf{d}\| > 0, 0 \leq \phi \leq \frac{\pi}{2}, \end{cases} \tag{3.4}$$

where k_0 is a constant set to alert for potential collisions, k_1 is the weight used for J, and k_2 is the weight used for f_G.

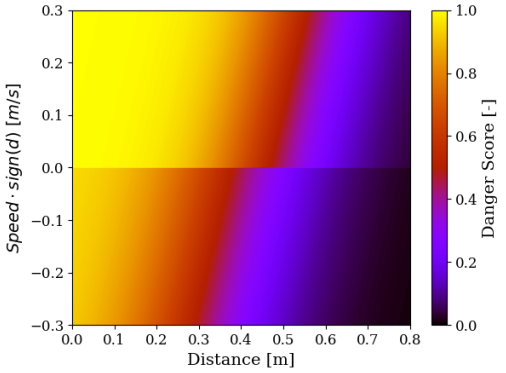


(a) $GPT_f = 1.0$

(b) $GPT_f = 1.0$



(c) $GPT_f = 0.5$



(d) $GPT_f = 1.5$

Figure 3.4: Relation of the Danger Score (DS) to the speed and distance for a custom case. sign(d) = 1 when $\phi > \frac{\pi}{2}$ and sign(d) = -1 otherwise. On the top left, the initial method proposed by [59]. On the top right, the proposed method with $GPT_f = 1$. On the bottom, from left to right, the proposed method with $GPT_f = 0.5$ and $GPT_f = 1.5$, respectively.

Figure 3.4 shows how the proposed DS behaves with different GPT_f compared to the original DS. Firstly, one can see that when the speed increases in the opposite

direction ($\phi < \frac{\pi}{2}$), the DS keeps decreasing instead of increasing, as shown in the original method. This is due to the imbalance in the weighted sum. Another distinctive feature is that even if the speed is near zero, the DS is not reduced to zero for short distances. Finally, using different GPT_f changes the behavior of the DS. Indeed, one can see that the distance at which the DS reaches 0.8 (high level of danger) is shifted to the left as GPT_f decreases. This means that increasing GPT_f increases the distance at which the DS starts to reach a high alert level. In other words, this can be interpreted as: "*The higher GPT_f is, the quicker the robot will be alert when moving toward the user.*"

3.5 Decision Making

This section describes the decision-making module proposed to assess the viability of utilizing LLMs for decision-making in HRC by providing a streamlined coding paradigm. Indeed, in robotics, it is often necessary to implement and think about all the possible situations in which the robot might end up and program the actions accordingly. By leveraging LLMs, one can describe the general attitude the robot should have with natural language and reduce the need for explicit condition descriptions. If the user wants the robot to behave in a certain way in specific situations, he only has to describe the situation and the desired outcome. If a situation has not been considered in advance, the LLMs can suggest an action by taking into account the general task and the other conditions described.

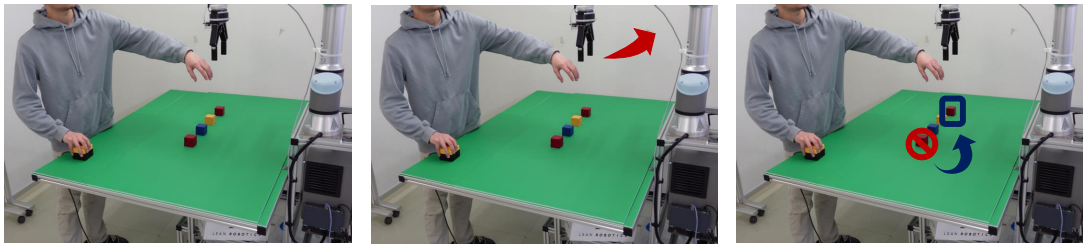
In this system, the decision-making process occurs when the robot's speed is reduced to zero because the safety index computed in Section 3.4 is too low. Indeed, whenever the DS exceeds 0.8, indicating a significant level of risk, the robot is programmed to halt its movement. This ensures that the robot stops whenever the situation poses a high degree of danger, safeguarding against unwanted collisions.

One way to decrease the DS is by asking the robot to move in the opposite direction of the obstacle (moving in the opposite direction influences the angle ϕ). This characteristic of the SI can be used to increase the efficiency and safety of the system by avoiding remaining stuck for an unnecessary amount of time. If a new objective in the opposite direction is suggested, the SI will increase and the robot will be able to move again. The robot has 3 functions he can choose from:

- h_1: the robot waits,
- h_2: the robot withdraws to an intermediate position,
- h_3: the robot changes the current objective.

In this module, GPT-4 determines the optimal function for the robot based on textual conditions provided by the user, as illustrated in Figure 3.5. The **first**

method, represented on the left, is a simple wait function (h_1) where the robot has to remain in its current position until the SI increases. The **second method**, represented in the middle, is a withdrawal method (h_2) where the robot moves to an intermediate location which is considered as safe and thus increases the distance with the user. After reaching this intermediate location, the robot tries again to go to the position he was going for previously. Moving to an intermediate position can make the situation safer without the user having to move and hopefully make the robot move around the previous obstacle. The **third method**, represented on the right, is a method that allows the robot to change objective (h_3). This method suggests aborting the current objective and changing to a new one instead of waiting for the situation to become safer or withdrawing to an intermediate position. Changing the objective can be advantageous because it can induce a new angle, which increases the SI. The implementation of the functions is detailed in Appendix 8.4.



(a) Wait (h_1) (b) Withdraw (h_2) (c) Change Objective (h_3)

Figure 3.5: Decisions available for the robot: Wait (h_1), Withdraw (h_2), and Change Objective (h_3).

The information provided to GPT-4 comprises three components: a description of the context, the robot's current state, and the user's input. The context description is explained in Appendix 8.3. The current state is based on proprioceptive and exteroceptive data. More precisely, it is composed of the following components:

$$F_{choice} = g(L_{in}, P_{O_i}, P_{init}, P_{\text{robot}}, P_{\text{human}}, A_{flag}, t_{\text{elapsed}}) \tag{3.5}$$

where L_{in} is the user natural language condition request, P_{O_i} is the Cartesian position of the remaining objects. P_{init} is the Cartesian position of the current goal, P_{robot} is the Cartesian position of the end-effector, P_{human} is the Cartesian coordinates of the human body parts, A_{flag} is a flag explaining whether the robot is picking or placing a cube and t_{elapsed} is the total time since the robot was immobilized.

Finally, the user can input the conditions he wants the robot to follow. For example, one can imagine a situation where the user's position changes a lot during

a collaborative task. In such a situation, the user could request the robot to wait since he is very likely to change position and get to a safer location in a short period of time. On the other hand, a user could be mostly static due to the nature of his task and it would be a waste of time for the robot to wait for the user to finish his task. The accuracy of the model and the emergent properties are explored in Section 4.4.

3.6 Robot Controller

This section describes the implementation of the methods introduced in Section 3.5. These methodologies are designed to demonstrate the potential of LLMs in guiding trajectories by suggesting three-dimensional (3D) poses based on multimodal input. The first function, h_1, is a rudimentary delay mechanism where the robot halts its actions until the SI increases. The decision-making process is reinitiated if the robot remains inactive after an additional two-second period. Three distinct withdrawal strategies are investigated for the second function, h_2. Lastly, the function h_3, which is responsible for changing objectives, is implemented utilizing GPT-4.

3.6.1 Withdrawal

Two methods named Modified Withdrawal and Full GPT Withdrawal are implemented to investigate alternative strategies for suggesting 3D withdrawal poses. The first method (Modified Withdrawal) aligns with the approach outlined by [72] and is augmented with the incorporation of an LLM to influence the impact of virtual forces. The second one (Full GPT Withdrawal) is independent of the method proposed by [72] as it solely relies on GPT-4 to suggest 3D poses. Unlike the original withdrawing method, they both use natural language to influence the direction toward which the robot must withdraw.

The initial method proposed by [72] uses a virtual force model to modify the end-effector velocity and move it not only away from the human but also toward a parking position (P_{parking}), which can be asserted as a safer location (*e.g.,* a position where the robot is out of reach for the user). The two virtual forces are represented in Figure 3.6. They can be computed with:

$$\begin{cases} \mathbf{F}^{\text{human}} = Mexp(\frac{-d}{R})\mathbf{n}, \\ \mathbf{F}^{\text{parking}} = A\mathbf{p}, \end{cases} \tag{3.6}$$

where M represents the magnitude of the force, R is its range, and d is the minimum distance between the human and the robot. The vector $\mathbf{n}$ is a normalized vector that describes the direction of the force. A is the magnitude of the force and $\mathbf{p}$ is the normalized vector pointing from the current location of the end-effector to the

parking position. The algorithm used to compute the new position can be found in Appendix 8.4.1.

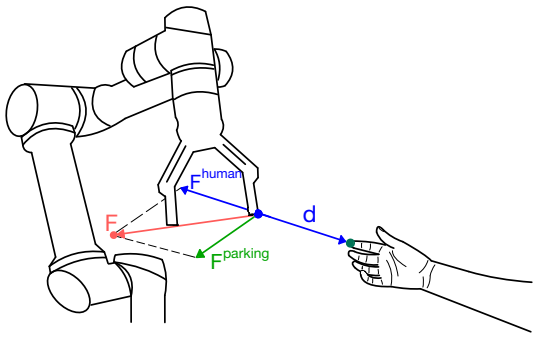


Figure 3.6: Representation of the virtual forces: $\mathbf{F}^{\text{human}}$ is a repelling force exerted by the human, $\mathbf{F}^{\text{parking}}$ is an attractive force exerted by the parking position and $\mathbf{F}$ is the resulting force. Figure inspired from [72].

The first method (Modified Withdrawal) presented in this thesis employs GPT-4 to adjust the parking position based on user input prompts. These prompts may specify either Cartesian changes relative to the user, such as "*Move to my left,*" or relative to objects, such as "*Move closer to the robot.*" By adapting the parking position accordingly, the robot gravitates towards a user-deemed safe position while adhering to a well-defined methodology with minimal probabilistic variability. The revised equation for the attractive forces can be formulated as:

$$\begin{cases} \mathbf{F}^{\text{human}} = Mexp(\frac{-d}{R})\mathbf{n}, \\ \mathbf{F}^{\text{parking}} = A\mathbf{p}(L_{in}), \end{cases} \tag{3.7}$$

where L_{in} is the input prompt provided by the user. The function $\mathbf{p}(L_{in})$ is implemented with GPT-4 by using the initial prompt detailed in Appendix 8.4.1. The equation for this method can be written as:

$$P_{obj} = h_2(L_{\text{context}}, P_{\text{parking}}, P_{\text{head}}, P_{O_i}, L_{in}) \tag{3.8}$$

where L_{context} is the context introduction regarding the task, P_{parking} is the current position of the parking position, P_{head} is the Cartesian position of the user (from which the commands are referred to), P_{O_i} is the Cartesian position of the objects and L_{in} is the user request.

The second method (Full GPT Withdrawal) diverges from the original withdrawal approach. In this method, GPT-4 directly suggests a new withdrawal position. To ensure the safety of the suggested position, GPT-4 is tasked with

providing a new position that maintains a specific angular relationship (ϕ_{lim}) with respect to the current position and obstacle direction, as illustrated in Figure 3.7. The angle ϕ_{new} refers to the angular disparity between the proposed withdrawal position and the current position-obstacle vector as defined in Equation 3.9.

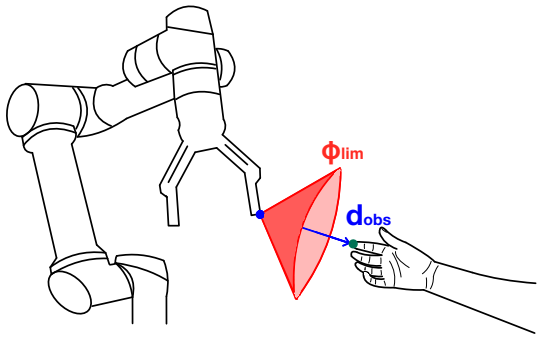


Figure 3.7: Representation of the angle ϕ_{lim} (in red), which must be smaller than ϕ_{new}. The closest point of the end-effector (P_{robot}) is depicted in blue, while the closest point of the human body, representing the obstacle (P_{human}), is shown in green. $\mathbf{d}_{obs}$ is the current position-obstacle vector.

$$\phi_{lim} < \phi_{new} = \arccos\left(\frac{\mathbf{d}_{obs} \cdot \mathbf{d}_{new}}{\|\mathbf{d}_{obs}\|\|\mathbf{d}_{new}\|}\right) \tag{3.9}$$

where $\mathbf{d}_{obs}$ is the current position-obstacle vector (in blue) as shown in Figure 3.7 and $\mathbf{d}_{new}$ is the current position-proposed withdrawal position vector. This method is implemented with GPT-4 by using the initial prompt detailed in Appendix 8.4.1. The equation for this method is written as:

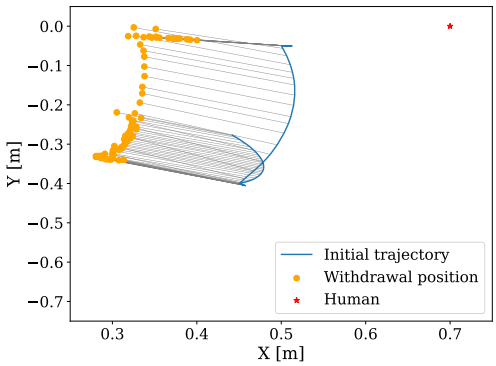
$$P_{obj} = h_2(L_{\text{context}}, P_{\text{robot}}, P_{\text{human}}, P_{\text{head}}, P_{O_i}, \phi_{lim}, L_{in}) \tag{3.10}$$

where L_{context} is the context introduction regarding the task, P_{robot} is the current position of the end-effector, P_{human} is the Cartesian position of the obstacle, P_{head} is the Cartesian position of the user (from which the commands are referred to), P_{O_i} is the Cartesian position of the objects (stacks, robot's base,...), ϕ_{lim} is the angle restriction and L_{in} is the user request.

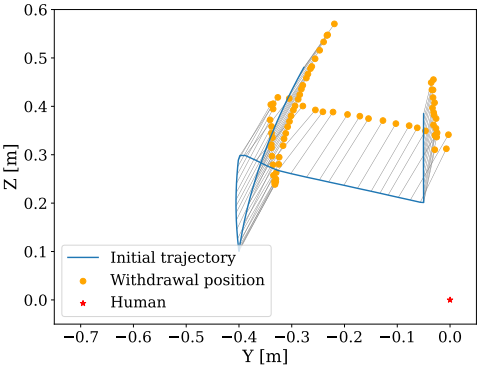
A comparison of the three withdrawal methods is provided in Figure 3.8. In blue is the original, real trajectory of the robot, in red is the position of the human closest body part, and in orange is the withdrawal position where the robot would go if it was asked to withdraw from its original trajectory with "*Withdraw on my left*" as user's input. Each row represents the withdrawal with a different method starting with the original withdrawal method proposed by [72] (original withdrawal).

Then, the method where GPT-4 modifies the parking position P_{parking} (Modified Withdrawal) and finally the method where GPT-4 directly suggests the withdrawal position (Full GPT Withdrawal).

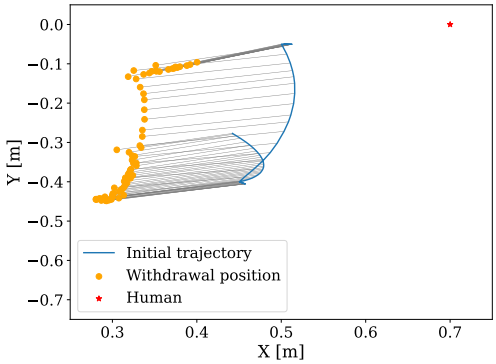
The Modified Withdrawal and the Full GPT Withdrawal both tend to move toward the left side of the user (the y coordinates of the withdrawal positions decrease) unlike the original withdrawal method. The Full GPT Withdrawal suggests a more discrete trajectory with widely spread out positions compared to the Modified Withdrawal.



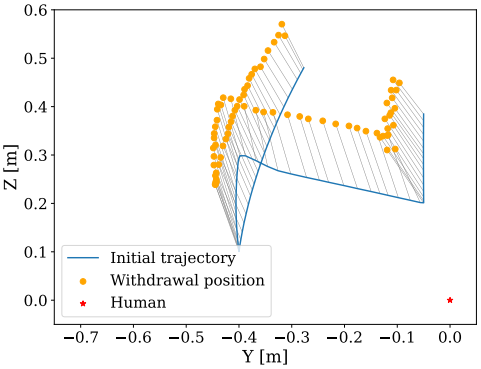
(a) Original Withdrawal



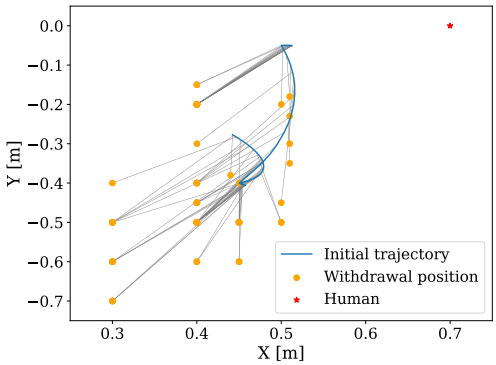
(b) Original Withdrawal



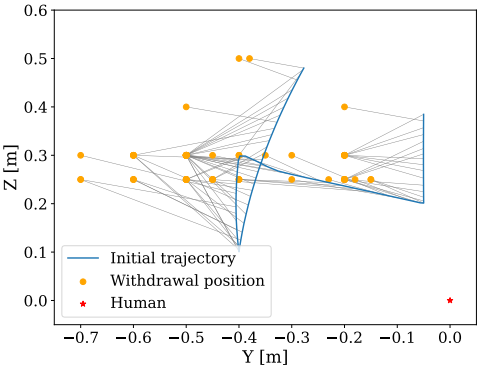
(c) Modified Withdrawal



(d) Modified Withdrawal



(e) Full GPT Withdrawal



(f) Full GPT withdrawal

Figure 3.8: Withdrawal positions compared to an initial trajectory if the user input is "*Withdraw on my left*": Original Withdrawal [72], Modified Withdrawal (changes P_{parking} from the Original Withdrawal with GPT-4), and Full GPT Withdrawal (GPT-4 suggests the withdrawal position). The position of the closest human body part is shown in red. The user's left side is defined by a decrease in the y-coordinate.

3.6.2 Changing Objective

This method is implemented to decide which object the robot should pick up based on its current situation and the user's natural language prompt. Equation 3.11 represents the function for this method:

$$P_{obj} = h_3(P_{\text{robot}}, P_{\text{head}}, P_{O_i}, L_{in}, P_{init}) \tag{3.11}$$

where P_{robot} is the Cartesian position of the end-effector of the robot, P_{head} is the Cartesian position of the user (from which the commands are referred to), P_{O_i} is the Cartesian position of the objects on the table and other relative objects (such as stacks, robot's base,...), L_{in} is the user request and P_{init} is the Cartesian position of the current objective the robot is moving toward. This method is implemented with GPT-4 by using the initial prompt detailed in Appendix 8.4.1.

Figure 3.9 depicts a decision made by the model for a simple example where the user requests the robot to take blocks that are far from him. On the left, the initial situation is represented and on the right, the block chosen by GPT-4 to be the next objective is highlighted in yellow.

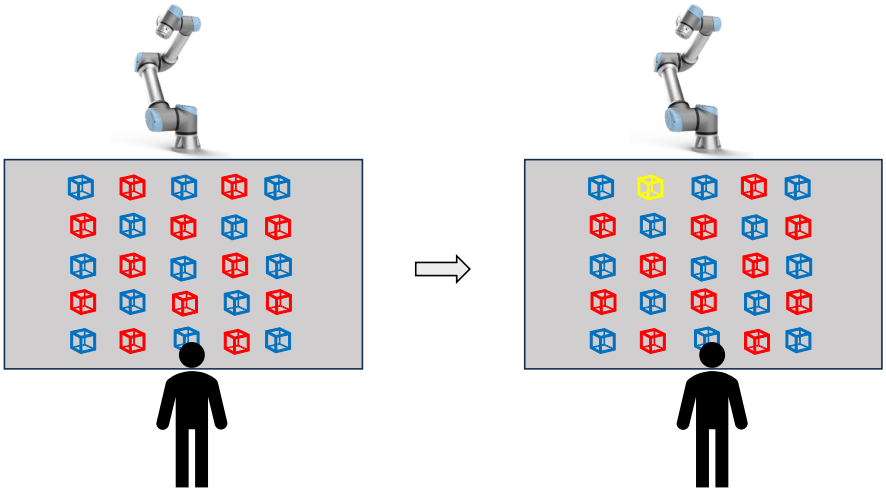


Figure 3.9: Representation of the function h_3 for a user input L_{in} = "*Take a block that is far from me.*" The left side depicts the initial state with all available objects. On the right, the object selected by GPT-4 for pickup is highlighted in yellow.

3.7 Constraint Satisfaction

This section focuses on the constraint satisfaction module, which ensures that the positions and speeds suggested by the system remain within acceptable ranges. The

inclusion of a constraint satisfaction module is imperative in the context of utilizing LLMs in robotics, particularly where safety is of the utmost concern. These models are prone to producing stochastic answers and occasional hallucinations. In such scenarios, the constraint satisfaction module serves as a critical safeguard, ensuring that the robot operates within predefined constraints and boundaries despite the inherent uncertainties. Once the objective coordinates P_{obj} are computed and processed by the constraint satisfaction module, the robot's path is computed and executed with a path planning algorithm.

The constraint satisfaction module governs both the output position (P_{in}) and a scaling factor (V_{in}) that influences the speed of the end-effector. Its operation varies depending on whether P_{obj} represents the position of an object or an intermediate pose. When P_{obj} corresponds to the position of an object, the submodule ensures the existence of the suggested object from GPT-4. This validation step is crucial for ensuring that the generated output is grounded in reality. Conversely, when P_{obj} represents an intermediate pose, the submodule verifies that this position is non-dangerous for the user and remains within the workspace of the robot. In addition to position constraints, the module imposes restrictions on the speed of the end-effector. This constraint scales the speed based on the SI, ensuring that the motion adapts to the danger of the situation.

When P_{obj} corresponds to the position of an object, P_{in} is computed with Equation 3.12.

$$P_{in} = \begin{cases} P_{obj} & \text{if } P_{obj} \in P_{O_i}, \\ P_{\text{robot}} & \text{if } P_{obj} \notin P_{O_i}, \end{cases} \tag{3.12}$$

where P_{robot} is the current position of the end-effector and P_{O_i} is a list of the positions of the objects in the scene. This means that the robot is simply going to wait if the position of the object given by GPT-4 is not in the scene. However, when P_{obj} represents an intermediate position, it is necessary to ensure that P_{obj} remains within the workspace with Equation 3.13:

$$P_{in} = \begin{cases} P_{obj} & \text{if } P_{obj} \in \mathcal{W}, \\ P_{\text{parking}} & \text{if } P_{obj} \notin \mathcal{W}, \end{cases} \tag{3.13}$$

where P_{parking} is a predefined safe parking position where the robot can withdraw and $\mathcal{W}$ is the designated safe workspace in which the robot can operate. If P_{obj} is indeed in the right workspace but is within the limited angle, the new position is computed with:

$$P_{in} = \begin{cases} P_{obj} & \text{if } \phi_{lim} < \phi_{new}, \\ P_{\text{parking}} & \text{if } \phi_{lim} > \phi_{new}, \end{cases} \tag{3.14}$$

where ϕ_{new} refers to the angular between the proposed withdrawal position P_{obj} and the current position-obstacle vector as defined in Equation 3.9. ϕ_{lim} refers to the limit at which the angle is not accepted as safe.

The constraint submodule is used to compute a scaling factor based on the safety index as shown in Equation 3.15:

$$V_{in} = m(SI) \tag{3.15}$$

where m is a function that computes a scaling factor based on the safety index (SI) which the path planning algorithm can then use.

Chapter 4

Experiments

The experiments aim to test hypotheses through the assessment of the methods outlined in Section 3. The experimental framework maintains uniformity across all trials, with consistent environmental conditions and tool utilization, as detailed in Section 4.1. The first experiment studies the integration of human safety perceptions into safety assessments by employing LLMs. The subsequent experiment delves into the accuracy and efficiency of a streamlined coding paradigm, leveraging the generalization capabilities of GPT-4 to apply conditions and make informed decisions regarding the robot's upcoming actions. Finally, the last experiment seeks to validate the potential of LLMs in shaping robotic arm trajectories by suggesting 3D poses influenced by multimodal input.

4.1 Experimental Setup

The experiments always comprise one user and one robotic arm. The user faces the robotic arm, separated by a table. On this table, there are objects that both the robot and the user can interact with during the experiment. The robot is equipped with a gripper to pick up these objects. Cameras are used to locate objects and human body parts. Figure 4.1 shows a CAD representation of the experimental setup. This standardized setup ensures consistency in each experiment, allowing for reliable comparative analysis and reproducibility of results.

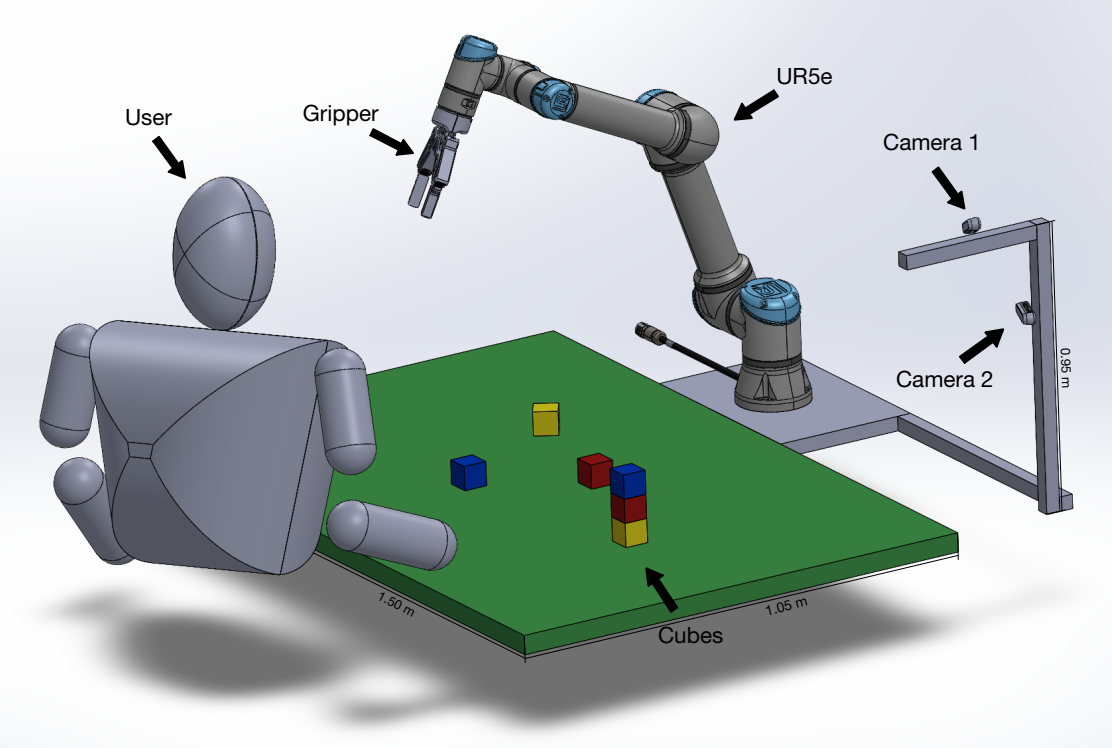


Figure 4.1: CAD representation of the experimental setup: the user faces the robot (UR5e) and they stack cubes together. Two cameras are placed on the side of the robot. One camera is used to detect P_{human} and the other is used to detect P_{O_i}.

The robot used for the experiments is the UR5e[1]. The UR5e is part of the Universal Robots' e-Series, which is designed to be collaborative with humans. It is known for its versatility, ease of use, and flexibility, making it suitable for a wide range of industrial applications such as assembly, machine tending, packaging, and quality inspection. It is equipped with built-in force sensing capabilities, allowing it to operate with precision and adapt to varying tasks and environments. The UR5e is a robotic arm with 6 revolute joints and has a cylindrical workspace. It can reach around itself in a cylindrical shape with the robot's base as the center. The UR5e has a reach of approximately 850 mm and a payload capacity of up to 5 kg. The end-effector in the experiments is a gripper[2]. The gripper kit is designed to be used with Universal Robots, specifically the UR series. It is a two-finger gripper with a maximum opening width of 140 mm.

During the different experiments, the robot has to pick up cubes on the table that are also accessible to the human user. The cubes are 3D printed and are all

[1]UR5e: https://www.universal-robots.com/products/ur5-robot/
[2]Gripper: https://robotiq.com/products/2f85-140-adaptive-robot-gripper

the same size (5 cm) with three distinct colors: blue, red, and yellow. The color of the object is not important for the following experiments but it could be used to further evaluate the understanding of GPT in different situations by expressing the danger each object represents. Other objects can replace the cubes.

RGB-D cameras are used to locate the human body parts (necessary for the assessment of safety) and the blocks on the table. Intel RealSense Depth Camera D435[3] are the cameras used in the experiments. They offer advanced depth sensing capabilities in a compact and versatile form factor, making them suitable for a robotic application. They are placed on the side of the robot. One camera is oriented toward the user to capture the upper body and accurately locate body parts while the second camera is directed at the table to identify the positions of the objects.

The system developed in this thesis is written and executed in a Software Development Environment (SDE) developed by [73]. The proposed SDE enables the collaborators to focus on their expertise and rely on automated tests and simulations from the system. It was initially designed for a robotic application which justifies its use in this thesis. The use of the SDE allows to further facilitate the use of important open source tools such as Robot Operating System (ROS)[4] or Docker[5]. This simplifies any extension to previous and future work related to this thesis. The SDE used for the following experiments uses ROS Noetic[6] with Ubuntu 20.04 on a Desktop PC.

Finally, the motion planning is computed with MoveIt[7] which is a framework designed to streamline the path planning and execution process for robotic systems. Leveraging state-of-the-art algorithms and seamless integration with ROS, MoveIt enables precise and efficient control of robot movements in various environments. In this system, MoveIt is provided with the final position and orientation of the end-effector P_{obj}. It computes and executes the motion planning given the system's constraints.

4.2 GPT Factor

The GPT factor experiment aims to demonstrate the capacity of LLMs to estimate human safety perceptions through a scaling factor. This is achieved by comparing the results from GPT-4 to the answers from a user study with the prompts listed in Table 5.1.

[3]RealSense D435: https://www.intelrealsense.com/depth-camera-d435/
[4]ROS: https://www.ros.org
[5]Docker: https://www.docker.com
[6]ROS Noetic: https://wiki.ros.org/noetic
[7]MoveIt: https://moveit.ros.org

4.2.1 Protocol

Participants of the experiment and GPT-4 are presented with the same initial prompt (Appendix 8.2) both for elucidating the scenario and specifying the desired output format. They then receive the different natural language inputs from the user. The answers from the participants are collected through a designated form.

4.2.2 Evaluation Method

Through a user study, participants are asked to suggest the GPT_f they find most appropriate for each prompt. The users' responses are then compared to the estimations provided by GPT-4. The mean of the users' responses and the answers from GPT-4 are used to compute the Root Mean Square Error (RMSE).

4.3 Safety Evaluation

The safety evaluation experiment aims to demonstrate the capacity of LLMs to integrate human safety perceptions into safety assessments. This is achieved by analyzing the robot's reactions and efficiency with different GPT_f across a spectrum of scenarios, ranging from easy to challenging. As explained in Section 3, the safety evaluation depends on GPT_f that scales the safety index proposed by [59].

This factor, ranging from 0 to 2, is influenced by user prompts. Extreme values (0 and 2) are reserved for scenarios in which the user intends to deactivate the robot's safety features or requires the robot to halt operation upon entry into the workspace. Intermediate values such as 0.5 and 1.5 represent heightened user comfort or discomfort around robots, respectively. These values are instrumental in evaluating the robot's performance under extreme conditions and are contrasted against the baseline safety index [59] obtained with a GPT factor of 1.0. Each safety index is evaluated through three distinct layouts.

These layouts are structured to simulate scenarios varying from non-challenging, where robot-human interaction is minimal, to moderately challenging, where workspace sharing is necessary, and finally to cooperative tasks requiring close collaboration between the robot and the user. Each layout is tested with each GPT factor as depicted in Figure 4.2.

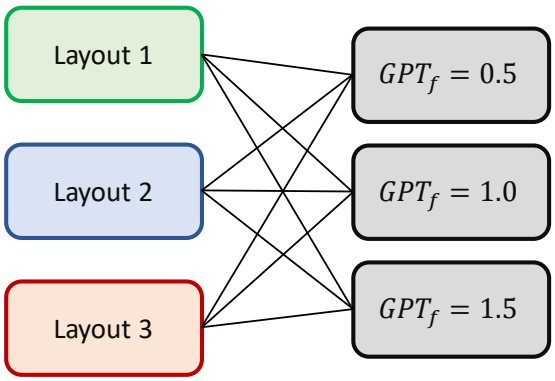
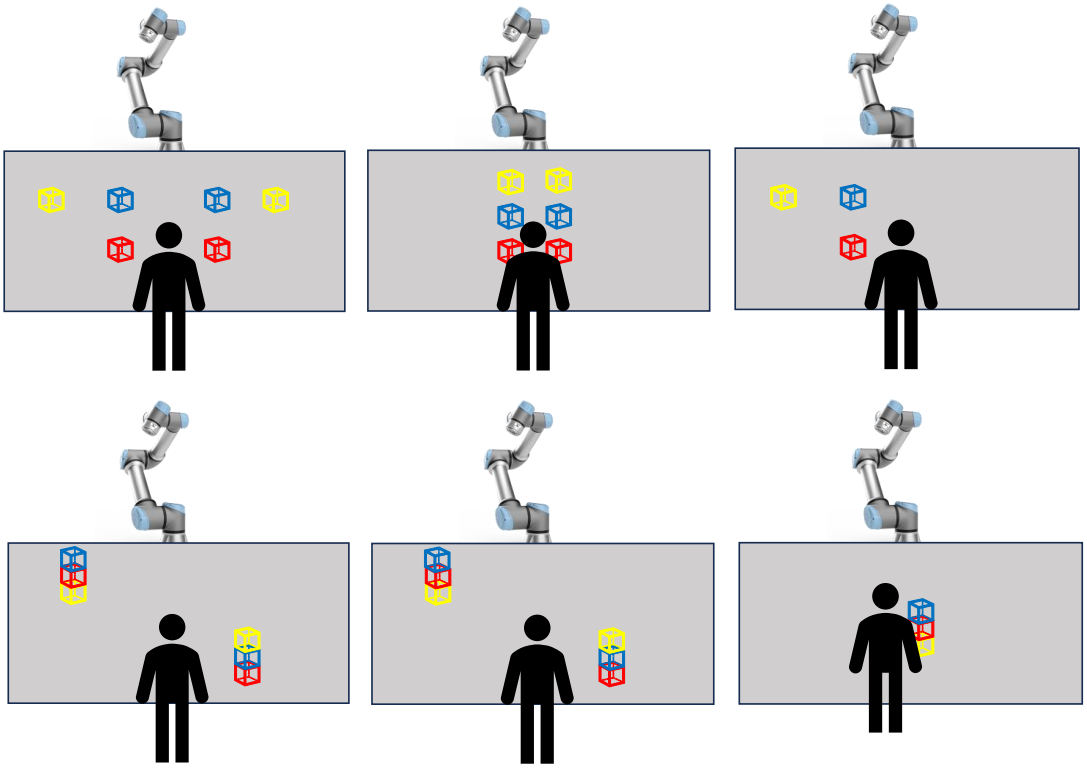


Figure 4.2: Each layout, which ranges from non-challenging (Layout 1) to challenging (Layout 3), is tested with three distinct scaling factors: $GPT_f = 0.5$, $GPT_f = 1.0$, and $GPT_f = 1.5$.

Figure 4.3 depicts the layouts designed for this experiment. In the first scenario, cubes are symmetrically positioned on either side of the workspace, requiring the robot and user to retrieve objects from their respective sides. Both parties are tasked with stacking the retrieved cubes on the same side as their initial location. While this arrangement minimizes potential overlap between the human and robot workspaces, a brief convergence exists when reaching for a centrally located cube. Nevertheless, collisions are anticipated to be avoided due to spatial planning.

The second scenario increases complexity by centralizing all cubes on the table. The user is directed to retrieve cubes from both sides, thereby increasing the likelihood of interaction between the human and robotic agents. Stacking procedures remain consistent with the first scenario. The spatial arrangement necessitates careful consideration of safety indices to mitigate collision risks during task execution.

In the third and most challenging scenario, the robot is tasked with retrieving and delivering cubes directly into the user's hand. This configuration represents a pivotal aspect of collaborative interaction, simulating scenarios where proximity between human and robot is imperative for task completion. The success of this task depends on an appropriate balance of safety thresholds: excessive caution (as dictated by overly conservative user prompts) may impede the robot's ability to effectively engage with the user. However, a user who expresses more confidence around robots will likely be able to achieve the task.



(a) Layout 1 (b) Layout 2 (c) Layout 3

Figure 4.3: Layout 1 represents the experimental setup considered as non-challenging, Layout 2 represents the experimental setup considered as moderate, and Layout 3 represents the experimental setup considered as challenging. The initial configuration of the scene is represented on the top, and the scene without interruption is represented on the bottom.

4.3.1 Protocol

When working with Layout 1 and Layout 2, the robot and the user must work at the same pace: when the robot starts reaching for the first cube, the user must start reaching for the first cube. Once the user reaches the cube, he must wait 3 seconds before initiating the placing motion. If the user finishes placing the cube first (which is likely since the robot tends to stop to protect the user), the user must wait for the robot to initiate the new picking motion to start reaching for the next cube. Algorithm 3 outlines the essential steps to ensure the experiment yields comparable results.

Algorithm 3 Protocol - Layout 1 and 2

```
while exp ≠ finished do                          ▷ Loop until experiment is finished
    Human_Pick()                                 ▷ Human picks up an object
    Human_Wait(3)                                ▷ Human waits for 3 seconds
    Human_Place()                                ▷ Human places the object
    while Robot_Placed() ≠ True do               ▷ Wait until the robot places its object
        Human_Wait()                             ▷ Human waits for the robot to place its object
    end while
end while
```

The third experiment is slightly different. After positioning his hand on the board, the user does not have to do anything.

Algorithm 4 Protocol - Layout 3

```
Human_Position()                                 ▷ Human places his hand forward
while exp ≠ finished do                          ▷ Loop until experiment is finished
    Robot_Pick()                                 ▷ Robot picks up an object
    Robot_Place()                                ▷ Robot places the object
end while
```

4.3.2 Evaluation Method

The evaluation is divided into a quantitative and a qualitative assessment. The quantitative assessment compares the results obtained with an SI produced by the scaling factors $GPT_f = 0.5$ and $GPT_f = 1.5$ to the default $GPT_f = 1.0$. The first measure compared is the distance between the nearest human body part and the robot end-effector, offering insights into the SI's behavior during collaborative tasks. For the same reasons, both the user's and the robot's speed are saved and analyzed. Finally, the average robot awareness (computed with the mean of the SI), the time required to finish the experiment, and task completion are all assessed.

Additionally, the qualitative assessment is done through a user study where the user is asked to evaluate within a 1-5 Likert scale the changes in the behavior of the robot. A video with the action of the robot and user with either $GPT_f = 0.5$ or $GPT_f = 1.5$ is compared to the same action with $GPT_f = 1.0$. For each video, the prompt used by the user to generate GPT_f is provided. Based on that natural language input, the user must evaluate his or her satisfaction with the changes in the robot's behavior.

4.4 Decision Making

An additional objective of the system is to assess the viability of utilizing LLMs for decision-making in HRC by providing a streamlined coding paradigm. This approach leverages the emergent properties of LLMs by diminishing the reliance on explicit condition descriptions and by using natural language to describe them. This approach is assessed using the decision-making module. When the DS surpasses a threshold of 0.8, the robot stops. Then, employing the method proposed in Section 3.5, GPT-4 makes an informed decision on the next action: whether to wait, withdraw, or change the objective. By providing additional input to the prompt, one can influence the decision-making process.

To assess this influence and evaluate its accuracy, the robot is halted in 24 different situations. These situations manipulate the input parameters necessary for the decision-making process. Text constraints imposed on input variables within the prompt allow the evaluation of decision-making accuracy by ensuring that the output aligns with these constraints. Three parameters vary to create the different situations:

- A_{flag}: the robot is picking or placing cubes,
- d: the distance between P_{robot} and P_{init},
- P_{O_i}: the layout of the cubes on the table.

In Figure 4.4, each unique combination of parameters is shown, resulting in a total of 24 experiments being conducted. On the right, one combination of the different parameters is represented: the robot is placing a cube within a distance shorter than 0.2m from its objective, and the experiment is in the layout 2 configuration. The distances between the end-effector of the robot and the closest point on the human body are categorized into three states: short where $d < 0.2$ m, middle where $0.2 \text{ m} < d < 0.4$ m, or long where $0.4 \text{ m} < d$.

The various layouts shown in Figure 4.5 aim to influence the decision of GPT-4 by varying the position and number of other blocks available for pick-up on the table. Layouts 1 and 2 (on the left) each have only two blocks, with one layout featuring blocks nearby and the other with blocks further apart. Layout 3 consists of only one objective left on the table. It is primarily used to evaluate the emergent properties of GPT-4 by ensuring it does not suggest changing the objective when there are no objectives left on the table without explicit instruction. Finally, the remaining Layout 4 has more objectives available on the table, allowing for a more complex decision-making process to be tested.

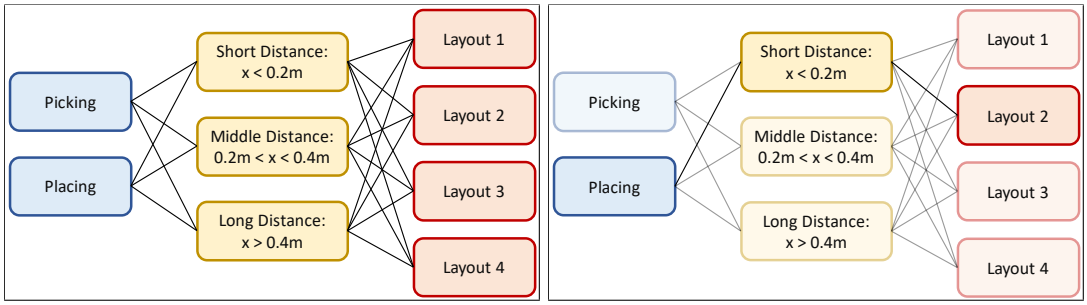
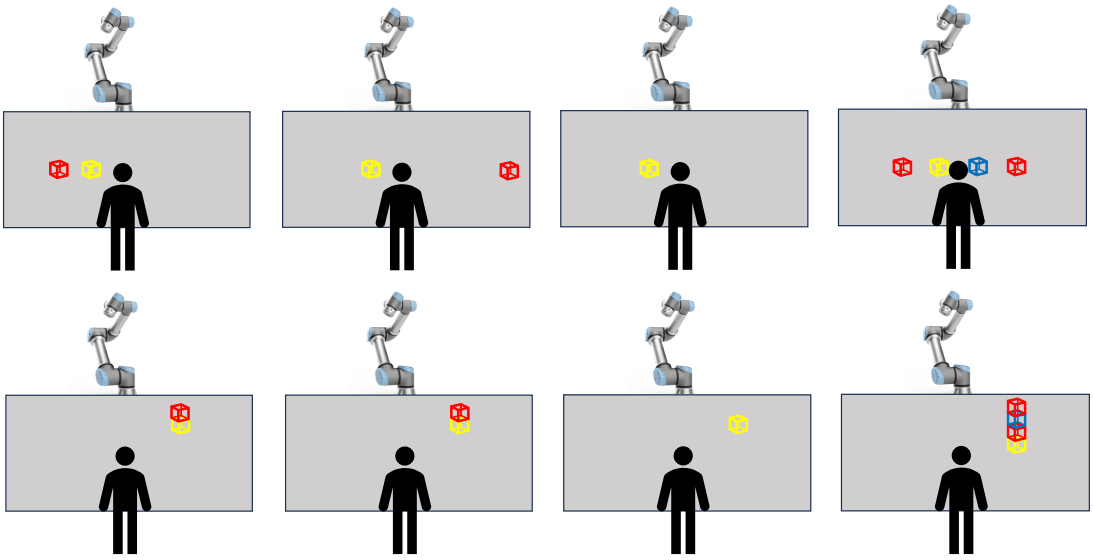


Figure 4.4: (Left) All possible combinations of parameters necessary for the decision-making process: in blue, the robot's action, in yellow, the distance between the end-effector and the human closest limb, in red, the layout of the cubes. (Right) Example of a possible combination of parameters used for the decision-making process: the robot is placing and its end-effector is stopped at less than 0.2 m from its current objective in the Layout 2.



(a) Layout 1 (b) Layout 2 (c) Layout 3 (d) Layout 4

Figure 4.5: Representation of the different layouts for the second experiment. The initial configuration of the scene is represented on the top, and the scene without interruption is represented on the bottom.

4.4.1 Protocol

To conduct the experiments, the user is required to maneuver a body part towards the robot within its designated stopping range, as determined by the variable

second parameter (short, middle, and long distance). Upon halting, the user is instructed to maintain their position for 2 seconds, allowing the robot to record its current state. The robot picks and places the predetermined cube toward a fixed position to ensure consistency across trials. The user must interrupt the robot's motion when it is advancing toward the yellow block, which serves as the robot's initial target for the picking motion. When the robot is placing, it consistently positions the yellow block toward a designated zone. Again, the user is asked to stop the robot by moving a body part close enough to the robot to make it stop when it reaches the assigned distance to its objective.

4.4.2 Evaluation Method

The potential use of GPT-4 as a streamlined coding paradigm across 24 different scenarios described previously is evaluated through a variety of user inputs. Three types of tests are employed to gauge the efficacy of this approach. Each test assesses balanced accuracy, execution time, and total token count to address the hypothesis.

The first test consists of paraphrasing the same condition on the input to assess its impact on the accuracy. The initial condition is: "*Always change objective when picking and always wait when placing. Never withdraw.*" This condition is then slightly modified to create new language inputs while preserving the original meaning. For example, an alternative formulation could be: "*Change the objective when picking and consistently wait when placing. Refrain from withdrawing.*" The list of the language inputs used can be found in Appendix 8.3. The 24 states are tested for each input prompt, and the resulting balanced accuracy is recorded.

Subsequently, different conditions are tested using language inputs that follow a similar structure. For instance, one condition could be: "*Always wait when picking. Otherwise always withdraw, Never change objective*" compared to another condition "*Always withdraw when picking. Otherwise always wait, Never change objective.*" Appendix 8.3 shows the complete list of conditions tested. Similar to the condition paraphrasing study, the 24 states are tested for each condition in the input prompt, and the resulting balanced accuracy is recorded.

Lastly, the method's scalability, which reflects the system's capability to handle more complex conditions, is assessed. These complex conditions entail combinations of the initial condition, such as the distance between the end-effector and the number of available blocks. For instance, compared to the initial condition "*Always change objective when picking and always wait when placing. Never withdraw*" where only the current action is conditioned upon, a more complex condition might be: "*When picking with 1 block left and distance with the objective is <0.2 m, change objective. Otherwise wait. Never withdraw.*" The full list of conditions tested can be found in Appendix 8.3. The 24 states are tested, and the resulting balanced accuracy is recorded.

Subsequently, three emergent properties are explored. The first emergent property explored is adaptability: without any prompt from the user, the robot shows a behavior that simulates an understanding of the context. Additionally, two prompts are used to evaluate how the model behaves when an input prompt is ambiguous or contradictory. Ambiguity is investigated by comparing the robot's results when no prompt is provided versus when a prompt with unclear conditions is given. Contradiction is assessed by comparing the robot's responses to a prompt containing contradictory conditions with its responses to no prompt.

4.5 Robot Controller

This experiment aims to evaluate the use of LLMs to shape robotic arm trajectories by suggesting 3D poses based on natural language input. The quality of the trajectories and 3D poses suggested are assessed by experimenting with the withdrawal and change of objective methods presented in Section 3.6. For the first experiment, the withdrawal method is tested with a variety of prompts. These prompts express positions relative to the user's location or other objects on the board. The second experiment, akin to the first, aims to gauge satisfaction with the change of objective based on user input. Each experiment is evaluated through a user base study of around 30 participants.

Figure 4.6, on the left, illustrates the initial trajectory pursued by the robot (depicted in blue). The task involves the robot picking up a cube positioned on the user's left side and relocating it between the user's hands. Throughout the experiment, the user's left hand serves as the closest point and thus represents the obstacle from which the robot attempts to withdraw. Trajectory points (without any withdrawal) are recorded along with the speed of the robot and safety indexes. Based on this data, three distinct trajectories are computed using the methods outlined in Section 3.6.1. Subsequently, two types of prompts are tested: commands relative to the user and commands relative to objects on the table. For instance, a command relative to the user might be: "*Move to the right,*" while a command relative to objects could be: "*Move closer to the stack.*"

The three withdrawal methods are tested for three coordinates along the trajectory for each prompt (9 trajectories per prompt). One at the beginning of the trajectory, one in the middle, and one at the end. On the right of Figure 4.6, one can see the three withdrawal trajectories for the point in the middle of the trajectory for the prompt "*Move to the left.*" Withdrawal 1 (in orange) is the original withdrawal method presented in [72], withdrawal 2 (in red) is the method where GPT-4 suggests the position of withdrawal, and withdrawal 3 (in green) is the method where GPT-4 suggests a new parking position.

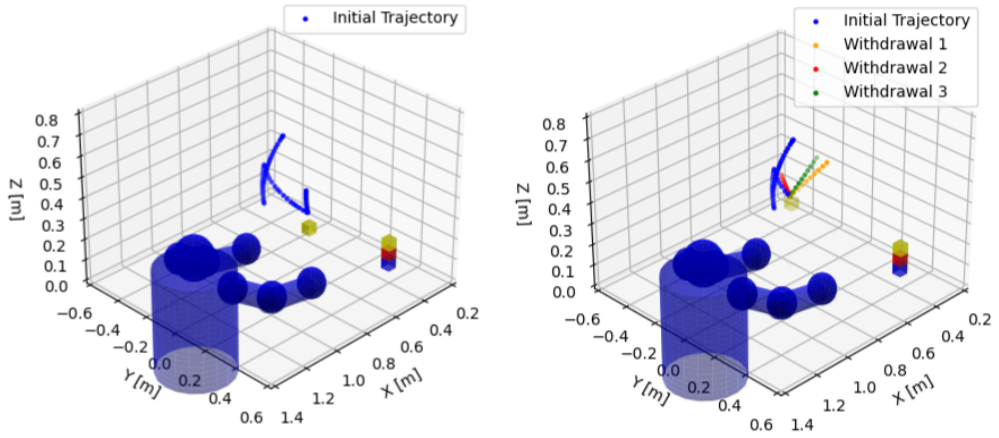


Figure 4.6: (Left) A representation of the initial trajectory without withdrawing. (Right) An example of the three withdrawals for the prompt "*Move to the left*" for an arbitrary point in the middle of the trajectory.

For the second experiment, the robot is stopped while picking up a cube. As depicted in Figure 3.9, there are 25 cubes on the table from which the robot can choose to change objective based on the user input. Similar to the first experiment, the user's input can consist of commands either relative to the user or relative to objects in the scene. For instance, a command relative to the user might be: "*Don't take cubes on the right,*" while a command relative to other objects could be: "*Take cubes that are close to the robot.*"

4.5.1 Protocol

In the first experiment, consistency is maintained by employing identical initial trajectories and user positions across all trials. Withdrawals based on this standardized trajectory are simulated to expand the pool of assessable data points and cases for participants. This approach ensures uniformity in experimental conditions and facilitates a thorough evaluation of diverse scenarios.

In the second experiment, simulated situations are utilized to enhance participant understanding of the situation and increase the richness of the dataset. Simulation allows for the generation of a greater number of data points, thereby enriching the analytical process.

In both instances, simulation offers the advantage of improved visualization of the input's impact, contributing to the overall quality and effectiveness of the experiments.

4.5.2 Evaluation Method

To answer the hypothesis, the first experiment is evaluated through a user study. For every withdrawal, combined with every coordinate and every prompt, the user answers the following question: "*Based on the request from the user, are you satisfied by the withdrawal trajectory proposed ?*" From which they need to choose an answer from a 1-5 Likert chart: "*Yes, very satisfied,*" "*Yes, satisfied*", "*Neutral,*" "*No, unsatisfied,*" and "*No, very unsatisfied.*" Then, they need to decide which withdrawal method they prefer. Finally, for every prompt, they need to justify what motivated their answers by selecting how significant the three factors were:

- Safety: the robot should always move away from the user.
- Compliance: the robot should always follow the command from the user.
- Human-like behavior: the robot should exhibit behavior that closely resembles that of a human.

Again, they need to evaluate each factor on a scale of 1 to 5 by selecting one of the following options: "*Very significant,*" "*Significant,*" "*Neutral,*" "*Low significance,*" and "*Not significant.*"

To further address the hypothesis, the second experiment is also assessed through a user study using 10 distinct prompts, all listed in Table 5.9. For each prompt, the user must choose which cube is more appropriate in his opinion for the robot to choose from.

Chapter 5

Results

This section presents both quantitative and qualitative results derived from the experiments detailed in Section 4. The initial set of results revolves around the integration of human safety perception into safety evaluations with the use of LLMs (GPT-4). Subsequently, the results show how viable LLMs are to provide a streamlined coding paradigm that can be used for decision-making in HRC. Finally, 3D poses influenced by natural language are examined, employing methods of withdrawal and objective alteration. These proposed poses are subject to evaluation through user studies.

5.1 Safety Evaluation

The results from the first experiment are divided into two parts: the analysis of GPT_f based on different prompts and the analysis of the robot's behavior for different GPT_f. The first part compares the GPT_f obtained with GPT-4 to GPT_f suggested by users through a user study. The second part then studies qualitative measures of robot behavior, such as completion time or the average speed of the end-effector for different GPT_f and layouts. It also studies the satisfaction of users through a user study.

5.1.1 GPT Factor

The complete prompt detailed in Appendix 8.2 is combined with the user feelings input to estimate GPT_f by leveraging GPT-4. Table 5.1 displays a non-exhaustive list of prompts that aims at showing how GPT_f varies depending on the input from the user. Each user input is also given to subjects who estimate GPT_f based on their perception of safety. The mean and the standard deviation of the answers from the user study are displayed in Table 5.1.

	Prompt	GPT-4	Mean	Std
P1	I feel very confident, give me everything you have	0.0	0.18	0.30
P2	I feel extremely confident around robots	**0.5**	**0.48**	0.25
P3	I feel confident around robots	0.8	0.60	0.23
P4	I am the best stack maker in the world	0.8	0.82	0.33
P5	I feel normal around robots	**1.0**	**1.01**	0.10
P6	I ate noodles yesterday	1.0	1.03	0.24
P7	I am heartbroken	1.5	1.25	0.31
P8	I feel uneasy around autonomous machines	1.2	1.32	0.30
P9	I am easily anxious	1.2	1.38	0.18
P10	I am not confident around robots	1.2	1.40	0.15
P11	I am very afraid of robots	**1.5**	**1.56**	0.34
P12	I am very afraid of robots, I don't want it to move	2.0	1.89	0.19

Table 5.1: Comparison of how GPT_f evolves depending on the input from the user when using GPT-4. It also shows the mean (Mean) and standard deviation (Std) of the answers provided by the user study.

The predictions from GPT-4 are compared to the mean of the user study by using the Root Mean Square Error (RMSE). $RMSE = 0.144$ shows the difference between the prediction from GPT-4 and the answers from the users. Figure 5.1 shows how the answers from GPT-4 and the subjects from the user study vary with the prompts from Table 5.1.

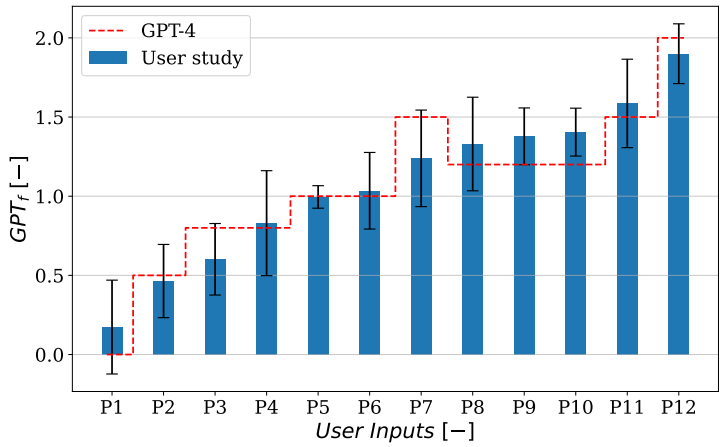


Figure 5.1: Evolution of GPT_f with different prompts: GPT_f computed by GPT-4 (in red) and the mean computed through the user study with its standard deviation (in blue). The user inputs named P1 to P12 represent the prompts from Table 5.1.

One can see the global trend of increasing mean with the different prompts which is similar for GPT-4. Another important point is that the GPT_f is usually higher than the mean computed in the user study when the user is more confident. This shows that the GPT-4 tends to be more alert than the users, which is usually desired for HRI systems. Finally, GPT-4 can emulate the understanding of the user's state of mind as it suggests a higher factor than neutral when the user is feeling heartbroken, for example.

5.1.2 Layout 1

This section presents the results for Layout 1, which is considered the easier experiment for the robot as detailed in Section 4.3. The cubes are symmetrically positioned on either side of the workspace, requiring the robot and user to retrieve objects from their respective sides. This arrangement minimizes potential overlap between the human and robot workspaces.

Figure 5.2 shows the evolution of the safety index and the distance between the user and the robot for Layout 1 with three distinct GPT_f. At the beginning of the experiment ($Time = 4$ s), the robot is going for the first cube and is located at a safe distance from the user (≈ 0.5 m). The picture on the top left shows the situation in which the user and the robot are at that moment with $GPT_f = 1.5$. At equal distances from the obstacle, the three SI exhibit a correlation with the lowest SI corresponding to the highest value of GPT_f. Conversely, the highest SI corresponds to the lowest GPT_f.

Around $Time = 20$ s, another difference emerges as $GPT_f = 0.5$ has a minimum distance with the user of 0.24 m compared to $GPT_f = 1.0$, which has a minimum distance of 0.4 m. The minimum distance for $GPT_f = 1.5$ is 0.45 m, which shows that when GPT_f increases, the robot stops earlier. This behavior is expected as the user, which requests a higher GPT_f through its language inputs, expects the robot to act more cautiously.

Finally, the time needed to finish the experiment increases with GPT_f. The two pictures on the right of Figure 5.2 show the situation of the robot and the user 50 s after the experiment started for $GPT_f = 0.5$ and $GPT_f = 1.0$. One can see that while $GPT_f = 0.5$ has already finished stacking the cubes, $GPT_f = 1.0$ is still around 10 seconds away from finishing.

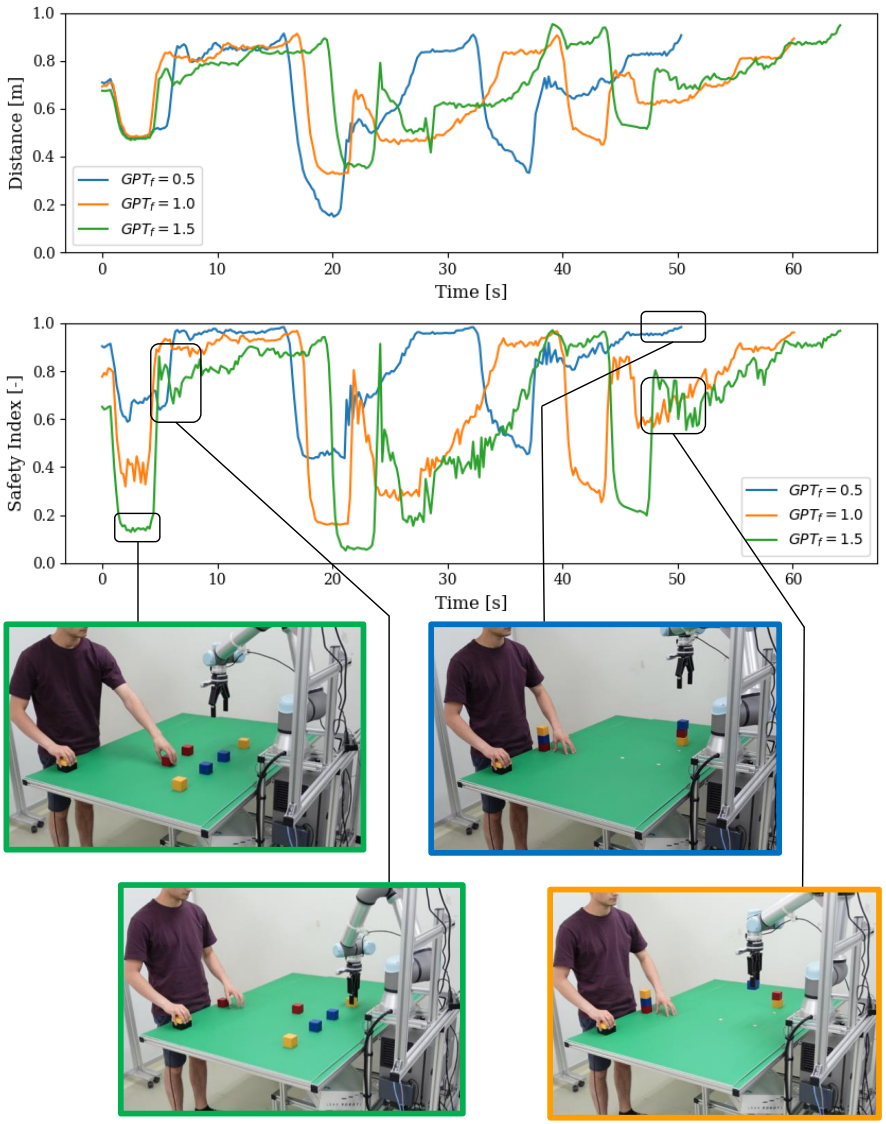


Figure 5.2: (Top) Distances between the robot end-effector and the closest human limb with two extreme GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.5$) and a baseline GPT factor $GPT_f = 1.0$. (Bottom) Safety Index (SI) during the same experiment with two extreme GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.5$) and a baseline GPT factor $GPT_f = 1.0$.

Figure 5.3 shows the robot's end-effector speed for the same experiment for $GPT_f = 0.5$ and $GPT_f = 1.5$. One can see that the robot's speed with $GPT_f = 1.5$ drops to zero once it gets closer to the first block compared to $GPT_f = 0.5$, which goes to the block without stopping. This behavior is the main reason for the overall time delay with a higher GPT_f. The second reason is the average speed of the

robot, with a higher GPT_f, the average speed of the robot is lower. Indeed, even at $Time = 1$ s, one can already see a difference in the speed of the robot. When the distance between the user and the robot is higher, the speed of the robot is similar no matter the GPT_f which is an expected behavior.

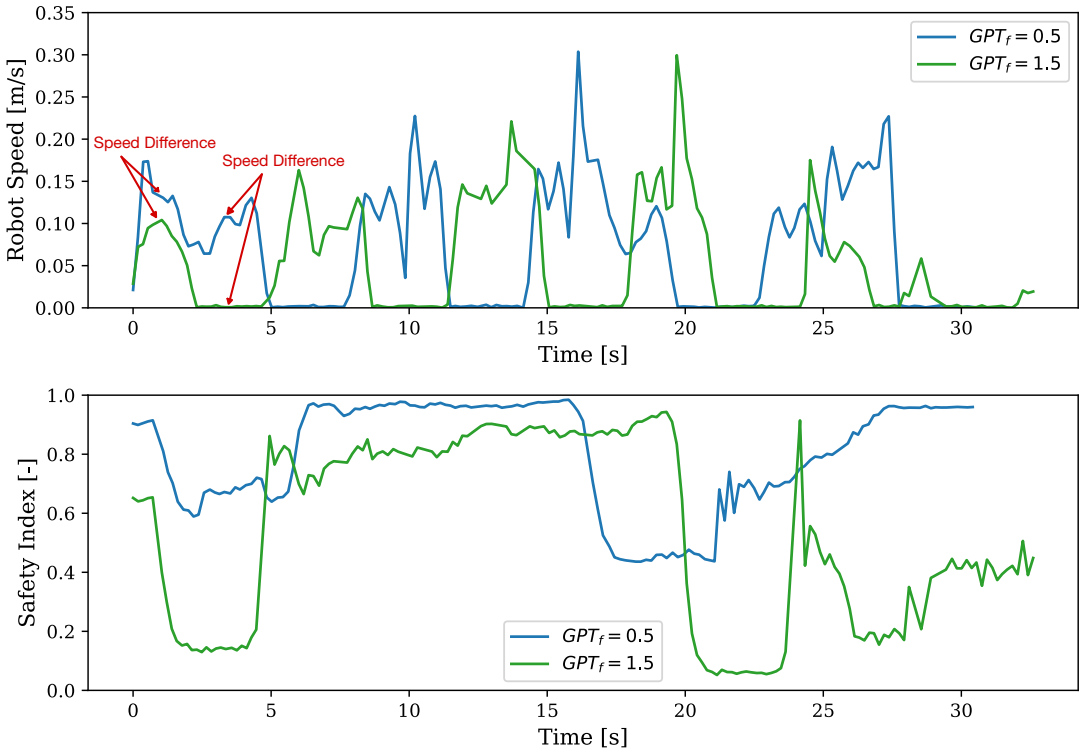


Figure 5.3: (Top) Speed of the robot end-effector with two extreme GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.5$). (Bottom) During the same experiment, Safety Index (SI) with two extreme GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.5$).

Table 5.2 shows the average distance between the robot end-effector and the user and the average speed. The trends in the table are consistent with those seen in Figure 5.3: the average speed increases as GPT_f decreases. Additionally, the average distance decreases with a reduction in GPT_f, which correlates with a shorter time required to complete the experiment.

	$GPT_f = 0.5$	$GPT_f = 1.0$	$GPT_f = 1.5$
Time [s]	50.31	60.11	64.12
Average Speed [m/s]	$0.749 \cdot 10^{-1}$	$0.619 \cdot 10^{-1}$	$0.600 \cdot 10^{-1}$
Average Distance [m]	$6.64 \cdot 10^{-1}$	$6.73 \cdot 10^{-1}$	$7.10 \cdot 10^{-1}$

Table 5.2: Comparison of the time needed to complete the experiment, the average speed of the end-effector, and the average distance between the end-effector of the robot and the closest human body part. All are compared with different GPT factors ($GPT_f = 0.5, 1.0$ and 1.5).

5.1.3 Layout 2

Layout 2 increases the complexity of the task by centralizing all cubes on the table as detailed in Section 4.3. The user is directed to retrieve cubes from both sides, thereby increasing the likelihood of interaction between the human and robotic agents.

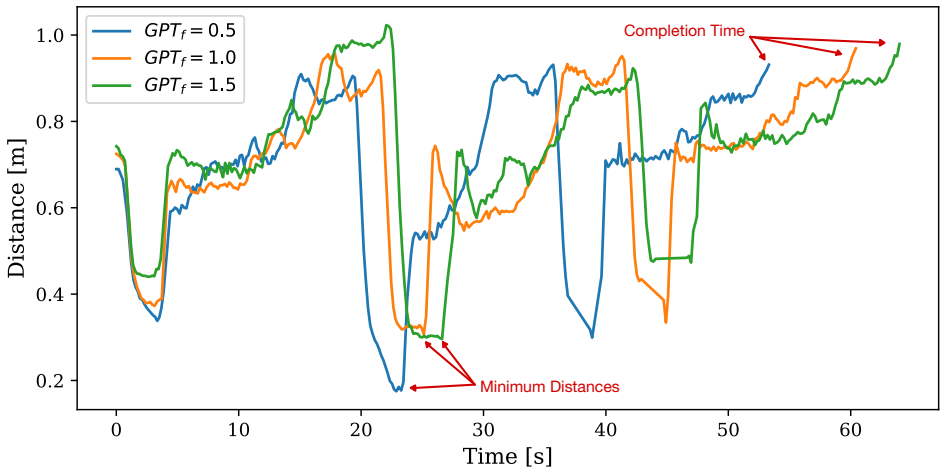


Figure 5.4: Distances between the robot end-effector and the closest human limb with two extreme GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.5$) and a baseline GPT factor ($GPT_f = 1.0$).

Similarly to the experiment with Layout 1, Figure 5.4 shows that the lower GPT_f is, the earlier the task is finished. Additionally, when $GPT_f = 0.5$, the distance between the robot and the user is smaller when they are reaching for a cube as the robot stops earlier for the other GPT_f. Table 5.3 shows similar properties as Layout 1 except for the average speed of the end-effector with $GPT_f = 1.0$ and

$GPT_f = 1.5$. This difference can be attributed to the experimental conditions, where slight variations in user position and timing between successive trials can influence the SI, which is directly related to the speed of the end-effector. The timing of the user's actions in reaching for a new cube may subtly impact the overall velocity of the robot.

	$GPT_f = 0.5$	$GPT_f = 1.0$	$GPT_f = 1.5$
Time [s]	53.32	60.42	63.96
Average Speed [m/s]	$0.758 \cdot 10^{-1}$	$0.668 \cdot 10^{-1}$	$0.669 \cdot 10^{-1}$
Average Distance [m]	$6.91 \cdot 10^{-1}$	$7.15 \cdot 10^{-1}$	$7.39 \cdot 10^{-1}$

Table 5.3: Comparison of the time needed to complete the experiment, the average speed of the end-effector of the robot, and the average distance between the end-effector and the closest human body part. All are compared with different GPT factors ($GPT_f = 0.5, 1.0$ and 1.5).

5.1.4 Layout 3

Layout 3 is the most challenging scenario as detailed in Section 4.3 since the robot is tasked with retrieving and delivering cubes directly into the user's hand. This configuration represents a pivotal aspect of collaborative interaction, simulating scenarios where proximity between humans and robots is imperative for task completion.

Layout 3 requires the robot to move in close proximity to the user without stopping, making the success of the task more challenging. Table 5.4 shows that $GPT_f = 0.5$ is the only one able to complete the task since it rarely stops due to the high nature of its SI. On the other hand, the two other factors would always stop before reaching their goal destination. When the robot was stuck for more than 20 seconds, the experiment was stopped, and "*" is placed next to the recorded value. A similar result was observed for $GPT_f = 1.5$, which explains its absence in Figure 5.5.

	$GPT_f = 0.5$	$GPT_f = 1.0$	$GPT_f = 1.5$
Time [s]	64.48	TIMEOUT	TIMEOUT
Average Speed [m/s]	$0.559 \cdot 10^{-1}$	$0.361 \cdot 10^{-1}$ *	$0.336 \cdot 10^{-1}$ *
Average Distance [m]	$4.58 \cdot 10^{-1}$	$4.77 \cdot 10^{-1}$ *	$5.47 \cdot 10^{-1}$ *

Table 5.4: Comparison of the time needed to complete the experiment, the average speed of the end-effector, and the average distance between the end-effector and the closest human body part. All are compared with different GPT factors ($GPT_f = 0.5, 1.0$ and 1.5). "*" means that the experiment timed out.

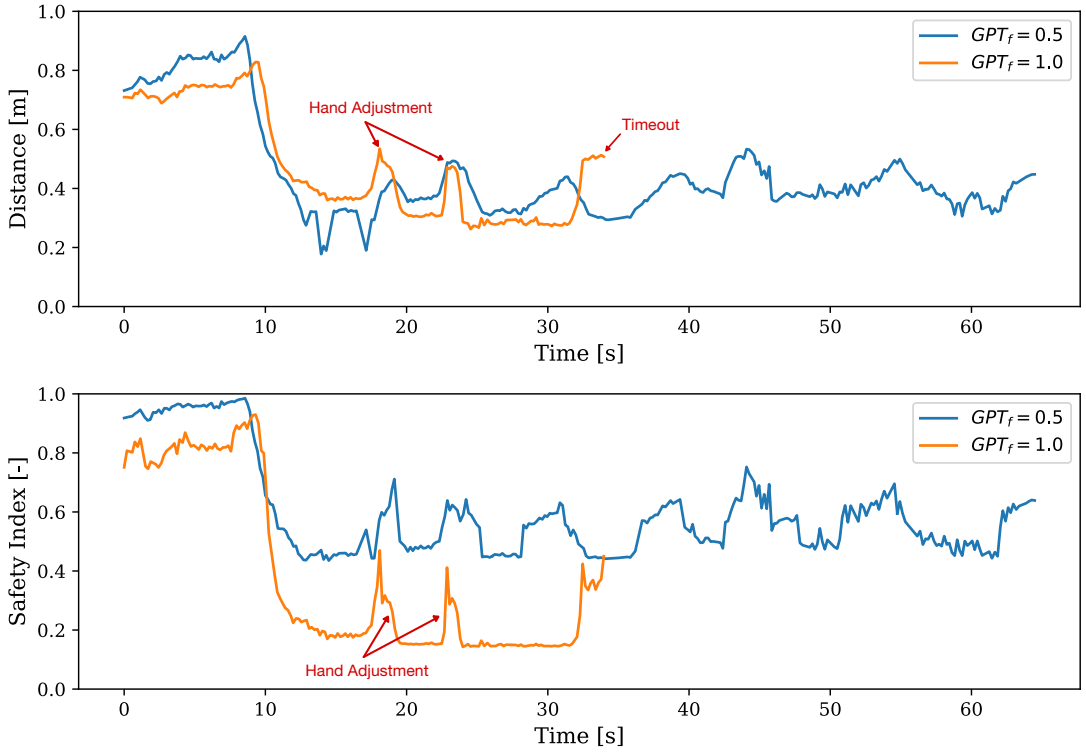


Figure 5.5: Distances between the robot end-effector and the closest human limb with two GPT factors ($GPT_f = 0.5$ and $GPT_f = 1.0$).

Figure 5.5 shows that the SI remains higher than 0.2 for the entire duration of the experiment with $GPT_f = 0.5$, which explains why the robot can complete the task. The same observation cannot be done for $GPT_f = 1.0$ as the SI is lower than 0.2 most of the time until the experiment is stopped. Some spikes in the SI and distance can be observed for $GPT_f = 1.0$ and they are caused by the user trying to remove his hand and placing it again closer to the robot in an attempt to be close enough for the robot to drop the cube. However, this attempt was unsuccessful as the robot did not finish the task.

These results show that not every collaborative task can be achieved with this method. Indeed, an inexperienced user who does not feel ready to use the robot to its full capabilities will not be able to complete this last task. This is desired because this task can be considered as more dangerous compared to the previous one.

5.1.5 Qualitative Results

In this qualitative assessment, a user study evaluates the user's satisfaction with the robot's behavior with different input prompts. They are requested to assess their level of satisfaction on a scale of 1 to 5 (1 is very unsatisfied and 5 is very satisfied) regarding the robot's new behavior in comparison to its neutral behavior (where $GPT_f = 1.0$).

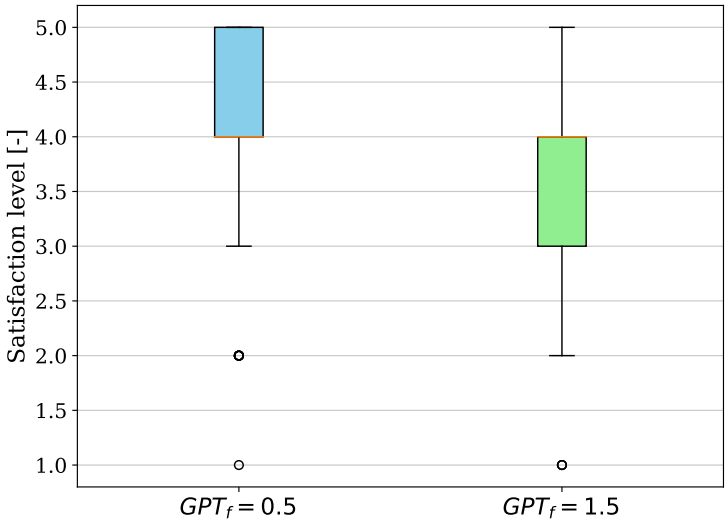


Figure 5.6: Satisfaction analysis from the user study for the impact of using $GPT_f = 0.5$ and $GPT_f = 1.5$ on the robot behavior compared to a neutral behavior ($GPT_f = 1.0$). 1 is very unsatisfied and 5 is very satisfied.

One can see that the satisfaction of both factors shares a similar median, but the satisfaction with $GPT_f = 1.5$ is slightly lower. Both distributions have a 75th percentile above 3, indicating that at least 75% of the responses for each factor are above the neutral satisfaction level of 3, showing generally positive assessments for both methods.

5.2 Decision Making

The results from the second experiment are divided into two parts: an analysis of balanced accuracy under different input conditions and an examination of emergent properties. The first analysis compares the expected output for a specific condition with the model's output when the condition is rephrased. Subsequently, it examines

the impact of altering the condition and the number of conditions. The second part compares the model's output under contradictory, ambiguous, or incomplete conditions.

5.2.1 Condition Assessment

As mentioned in Section 4, three distinct tests are made to evaluate the viability of using LLMs as a streamlined coding paradigm. The first one consists of changing the natural language used to express the same condition on the input and thereby assessing the **condition paraphrasing**. Then, with language inputs that follow a similar structure, namely **condition adjustment**, different conditions are tested. The final step is to assess the scalability of the condition (**condition scalability**), gauging the system's capability to effectively manage increasingly intricate conditions.

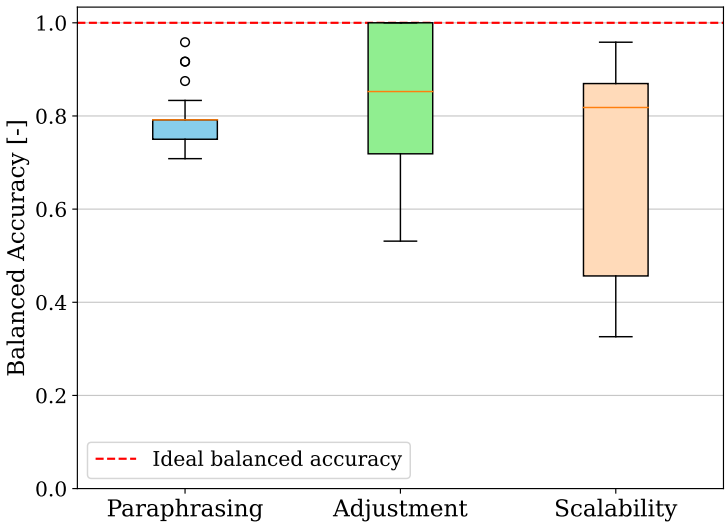


Figure 5.7: Balanced accuracy of the model with three variable parameters: condition paraphrasing (the same condition is expressed with different words), condition adjustment (the condition on the input is changed), and condition scalability (additional constraints on the input data).

The difference in balanced accuracy for the three parameters is illustrated in Figure 5.7. Condition paraphrasing shows the smallest standard deviation with a median slightly below the other medians. The condition adjustment displays a notably dispersed distribution, as shown by its higher standard deviation, with

multiple balanced accuracy measurements nearing the 50% threshold. This dispersion is effectively depicted in the boxplot, where the lower quartile extends toward approximately 50%. Condition scalability demonstrates elevated variance compared to condition adjustment, indicating that as the number of conditions increases, so does the variance.

	Lexical variation	Condition adjustment	Condition scalability
Balanced Accuracy [-]	0.79	0.85	**0.70**
Execution Time [s]	0.82 ± 0.50	0.83 ± 0.46	0.87 ± 0.50
Tokens [-]	649.5 ± 54.0	658.4 ± 54.3	660.2 ± 54.7

Table 5.5: Comparison of the balanced accuracy (mean), execution time, and number of tokens used for completion for three variable parameters: lexical variations (the same condition is expressed in different ways), condition adjustment (the condition on the input is changed), and condition scalability (additional constraints on the input data).

It should be noted that the execution time experiences a slight uptick when additional constraints are imposed on the input data compared to the other parameters. Conversely, the number of tokens remains relatively stable across conditions, indicating consistent lengths of expressions required to formulate the varied equations. However, it is worth noting that despite the minor differences in execution time, these increments might accumulate in scenarios demanding real-time or high-throughput processing, warranting consideration in robotic applications that require real-time such as HRC on a production line.

5.2.2 Emergent Properties

As detailed in Section 4, three emergent properties are explored. The first emergent property explored is adaptability: without any prompt from the user, the robot shows a behavior that simulates an understanding of the context. Additionally, two prompts are used to test what happens when the prompt is ambiguous and contradictory. Ambiguity is investigated by comparing the robot's results when no prompt is provided versus when a prompt with unclear conditions is given. Contradiction is assessed by comparing the robot's responses to a prompt containing contradictory conditions with its responses to no prompt.

Table 5.6 presents the model outputs without imposing specific constraints on user inputs. These outputs correspond to the 24 distinct states detailed in Section 4. The first column indicates the current action of the robot, with 1 denoting picking and 2 denoting placing. The second column describes the distance between the

end-effector and the robot's current objective, categorized as follows: 1 for distances smaller than 0.2m, 2 for distances between 0.2m and 0.4m, and 3 for distances exceeding 0.4m. The third column illustrates the layout of cubes on the table, ranging from zero cubes to 4 cubes. Finally, the last column displays the output generated by GPT-4, with 1 indicating the robot's decision to wait, 2 deciding withdrawal, and 3 ordering a change in objective.

Action	**Distance**	**Layout**	**Answer**
Picking	x < 0.2 m	1	Wait
Picking	x < 0.2 m	2	Wait
Picking	x < 0.2 m	3	Wait
Picking	x < 0.2 m	4	Wait
Picking	0.2 < x < 0.4 m	1	Change Objective
Picking	0.2 < x < 0.4 m	2	Change Objective
Picking	0.2 < x < 0.4 m	3	Withdraw
Picking	0.2 < x < 0.4 m	4	Change Objective
Picking	0.4 m < x	1	Change Objective
Picking	0.4 m < x	2	Change Objective
Picking	0.4 m < x	3	Withdraw
Picking	0.4 m < x	4	Change Objective
Placing	x < 0.2 m	1	Wait
Placing	x < 0.2 m	2	Wait
Placing	x < 0.2 m	3	Wait
Placing	x < 0.2 m	4	Wait
Placing	0.2 < x < 0.4 m	1	Withdraw
Placing	0.2 < x < 0.4 m	2	Withdraw
Placing	0.2 < x < 0.4 m	3	Withdraw
Placing	0.2 < x < 0.4 m	4	Withdraw
Placing	0.4 m < x	1	Withdraw
Placing	0.4 m < x	2	Withdraw
Placing	0.4 m < x	3	Withdraw
Placing	0.4 m < x	4	Withdraw

Table 5.6: General results from the prompt used in Appendix 8.3 for the 24 states possible without additional conditions. Cells colored in blue indicate the answers that show noteworthy results.

The first sign of emergent properties is apparent with the initial prompt. As depicted in Table 5.6, the robot alters its objective when it is more than 0.2 m away from its current objective (0.2 m < x) while picking objects (in blue). However,

this behavior does not occur when no other objectives are available on the table. Remarkably, GPT-4 autonomously refrains from changing its objective when no new objective is available, even without explicit instruction to do so. This behavior holds significance, particularly in scenarios where users may not wish to explicitly outline all potential situations the robot could encounter.

Action	**Distance**	**Layout**	**Answer**
Picking	x < 0.2 m	1	Wait
Picking	x < 0.2 m	2	Wait
Picking	x < 0.2 m	3	Wait
Picking	x < 0.2 m	4	Wait
Picking	0.2 < x < 0.4 m	1	Change Objective
Picking	0.2 < x < 0.4 m	2	Change Objective
Picking	0.2 < x < 0.4 m	3	Wait
Picking	0.2 < x < 0.4 m	4	Change Objective
Picking	0.4 m < x	1	Change Objective
Picking	0.4 m < x	2	Change Objective
Picking	0.4 m < x	3	Withdraw
Picking	0.4 m < x	4	Change Objective
Placing	x < 0.2 m	1	Wait
Placing	x < 0.2 m	2	Wait
Placing	x < 0.2 m	3	Wait
Placing	x < 0.2 m	4	Wait
Placing	0.2 < x < 0.4 m	1	Wait
Placing	0.2 < x < 0.4 m	2	Wait
Placing	0.2 < x < 0.4 m	3	Wait
Placing	0.2 < x < 0.4 m	4	Wait
Placing	0.4 m < x	1	Withdraw
Placing	0.4 m < x	2	Withdraw
Placing	0.4 m < x	3	Withdraw
Placing	0.4 m < x	4	Withdraw

Table 5.7: Results from the prompt "*Wait if action is ongoing*" for the 24 states possible with an ambiguous condition. Cells colored in blue indicate the answers that show noteworthy results.

Another emergent phenomenon arises from the prompt "***Wait if action is ongoing***" that lacks explicit instructions regarding when the robot should wait. Consequently, it remains unclear whether the user intends for the robot to wait during picking, placing, or both stages. Table 5.7 shows a noticeable trend observed

in GPT-4 responses with a tendency toward more frequent instances of waiting. This trend indicates GPT-4's grasp of the significance of waiting, a departure from conventional programmed conditions where such nuanced decision-making is unattainable.

Action	**Distance**	**Layout**	**Answer**
Picking	x < 0.2 m	1	Wait
Picking	x < 0.2 m	2	Wait
Picking	x < 0.2 m	3	Wait
Picking	x < 0.2 m	4	Change Objective
Picking	0.2 < x < 0.4 m	1	Change Objective
Picking	0.2 < x < 0.4 m	2	Change Objective
Picking	0.2 < x < 0.4 m	3	Withdraw
Picking	0.2 < x < 0.4 m	4	Change Objective
Picking	0.4 m < x	1	Change Objective
Picking	0.4 m < x	2	Change Objective
Picking	0.4 m < x	3	Change Objective
Picking	0.4 m < x	4	Change Objective
Placing	x < 0.2 m	1	Wait
Placing	x < 0.2 m	2	Wait
Placing	x < 0.2 m	3	Wait
Placing	x < 0.2 m	4	Wait
Placing	0.2 < x < 0.4 m	1	Withdraw
Placing	0.2 < x < 0.4 m	2	Withdraw
Placing	0.2 < x < 0.4 m	3	Withdraw
Placing	0.2 < x < 0.4 m	4	Withdraw
Placing	0.4 m < x	1	Withdraw
Placing	0.4 m < x	2	Withdraw
Placing	0.4 m < x	3	Withdraw
Placing	0.4 m < x	4	Withdraw

Table 5.8: Results from the prompt "*When you are closer than 20cm from your objective, you must wait. With any other distance between the end-effector and the objective, you must Withdraw. When you are picking, you must always change objective.*" for the 24 states possible with a contradictory condition. Cells colored in blue indicate the answers that show noteworthy results.

The prompt "***When you are closer than 20cm from your objective, you must wait. With any other distance between the end-effector and the objective, you must Withdraw. When you are picking, you must always change objective.***" illustrates GPT-4's response when confronted with

contradictory conditions. The contradiction arises from the prompt's directive for the robot to either withdraw or wait based on distance, while simultaneously instructing the robot to alter the objective during cube selection. The inherent challenge of this prompt stems from the concurrent execution of the picking action and the distance assessment. Consequently, the robot may be in the process of picking an object while also being within the prescribed distance threshold of the objective, leading to a contradiction in the decision between waiting or changing the objective.

Table 5.8 shows that the answers from GPT-4 remain largely unchanged compared to the initial prompt, with only two answers exhibiting shifts, likely influenced by the final sentence in the prompt. This implies that GPT-4 might overlook contradictory conditions, diverging from traditional programmed conditions where contradictions lead to undesired outcomes or errors.

5.3 Robot Controller

The results from the third experiment are segregated into two sections, both derived from a user study: an evaluation of withdrawal methods and an assessment of the change of objective method. Initially, the analysis contrasts user satisfaction levels regarding various withdrawal methods outlined in Section 3. Subsequently, it delves into the factors that influenced the users' reasoning. In the second part, the model's output regarding the object to be picked up by the robot is assessed using responses obtained from the user study.

5.3.1 Withdrawal

The methods' performance was evaluated in a user study, where around 30 participants assessed a total of 30 videos. Each participant was asked to rate the withdrawal methods on a 1-5 Likert scale based on a given natural language input.

Figure 5.8 summarizes the distribution of the answers for each method. Withdrawal 1 represents the withdrawal method proposed by [72], Withdrawal 2 is the same method with the parking position modified by GPT-4, and Withdrawal 3 is the method where GPT-4 directly suggests a new withdrawal position. As depicted in Figure 5.8, Withdrawal 3 received the highest satisfaction ratings and the lowest dissatisfaction ratings. Withdrawal 1 and Withdrawal 2 show a smaller difference in their results.

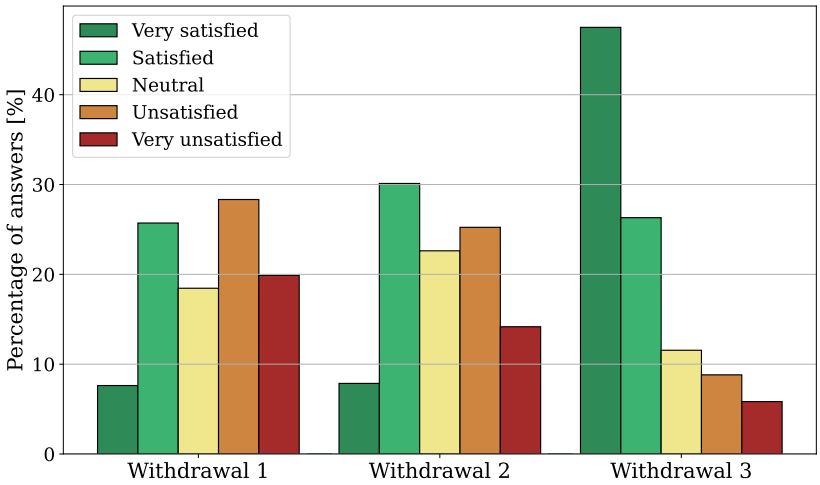


Figure 5.8: User study distributions of answers for each withdrawal method. Withdrawal 1 (Original Withdrawal) is the original withdrawal method proposed by [72], Withdrawal 2 (Modified Withdrawal) is the original withdrawal method augmented with the parking position modified by GPT-4, and Withdrawal 3 (Full GPT Withdrawal) is the withdrawal method where GPT-4 directly suggests a withdrawal position.

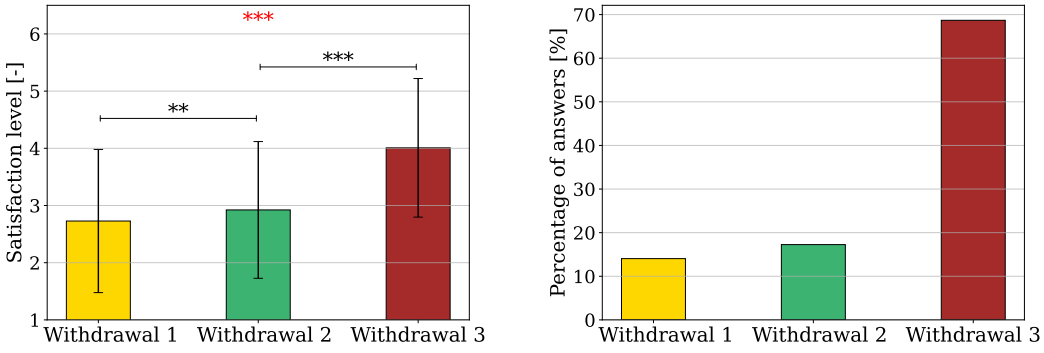


Figure 5.9: (Left) Satisfaction level of the three withdrawal methods ranging from 1 (very unsatisfied) to 5 (very satisfied). ***: $p < 0.001$, **: $p < 0.01$. (Right) Percentage of answers for the preferred withdrawal method.

In the same user study, participants were asked to select their preferred withdrawal method if they were in the user's situation. Figure 5.9, on the left, displays the average satisfaction level, ranging from 1 (very unsatisfied) to 5 (very satisfied). A Kruskal–Wallis test and two Mann–Whitney U tests were conducted to examine

the differences between the means and confirm whether the preference for the proposed method indeed favored Withdrawal 3. The p-values for Withdrawal 1 and 2 are lower than 0.01 but higher than 0.001, unlike the p-value between Withdrawal 2 and 3, where the null hypothesis was strongly rejected. In the same user study, participants were asked to select their preferred withdrawal method if they were in the user's situation. As shown on the right of Figure 5.9, users favored Withdrawal 3 with Withdrawal 1 and 2 still chosen in some cases. When Withdrawal 3 was not chosen as their preferred withdrawal method, Withdrawal 2 was chosen more often than Withdrawal 1.

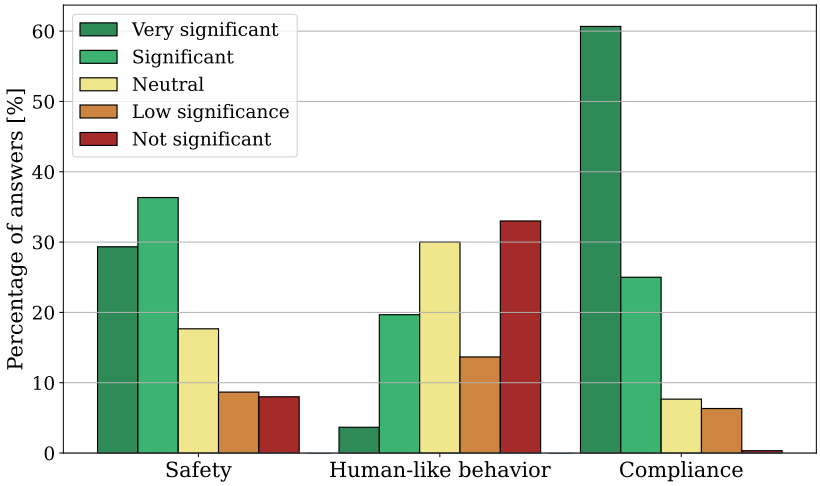


Figure 5.10: User study distributions of answers for each factor influencing the answers. Safety means that it is important for the user that the robot always moves away from the user. Compliance means that the robot should always follow the user's command. Human-like behavior means that the robot should exhibit behavior that closely resembles that of a human.

The users were also asked to assess the significance of each factor for their previous answers. Figure 5.10 displays the distribution of their responses. They had to evaluate three factors: safety, which refers to the robot moving to a safer location; compliance, which indicates the robot following the user's command; and human-like behavior, which involves the robot exhibiting behavior resembling that of a human. According to the users' assessments, the most significant factor for them was compliance, followed by safety. Human-like behavior was generally deemed not significant by the users.

Finally, the distributions from the users were divided into three groups: students from UCLouvain who did not receive an engineering education (UCLouvain-Other), students from UCLouvain who did receive an engineering education (UCLou-

vainEngineer), and students from Ritsumeikan University who received an engineering education (Ritsumeikan-Engineer). UCLouvain students study in Belgium, while Ritsumeikan students study in Japan. Figure 5.11 illustrates the significance level for the factors impacting their decision and the preference for the withdrawal methods are similar across all three groups.

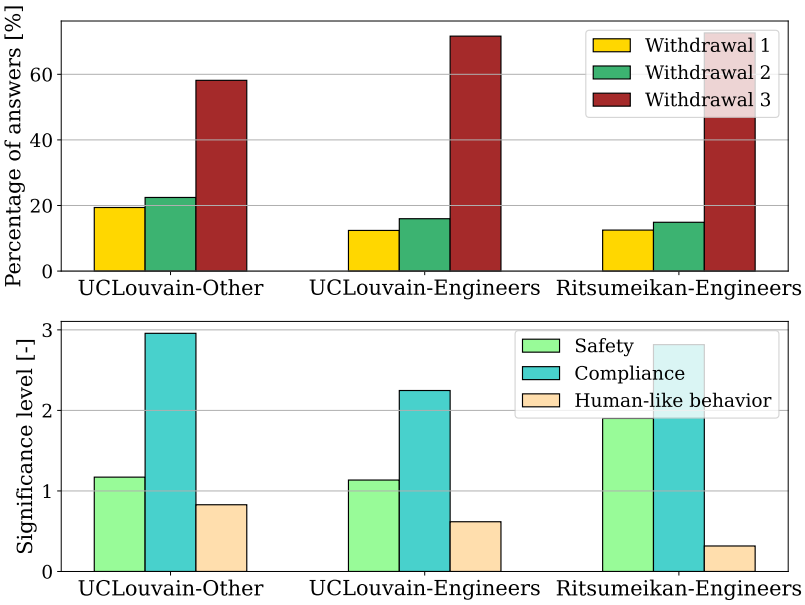


Figure 5.11: User study distributions of answers for each withdrawal method and factors influencing the answers. Distributions are divided into three subgroups: UCLouvain-Other (UCLouvain students with no engineering background), UCLouvain-Engineers (UCLouvain students with engineering background), and Ritsumeikan (Ritsumeikan students with engineering background).

5.3.2 Changing Objective

The change in objective is evaluated by comparing the responses provided by GPT-4 and users in a user study when given the same instructions, such as "*Pick a block between me and the robot.*" Figure 5.12 illustrates how the results are plotted relative to the experiment scene. On the left side, the position of the user (referred to as "*Me*" or "*User*" in the prompts) is depicted with respect to the cubes on the table. The robot and its position are also indicated at the top of the board. The yellow-colored block represents the block selected by GPT-4. On the right side, the representation is used to display the responses from the user study. The block chosen by GPT-4 is colored in red, and the frequency (f_{gt_i}) indicates the number of times a user chose this block relative to the maximum number of times any block

was chosen. Each circle represents one cube on the diagram on the left.

The error of the answer suggested by GPT-4 for each case can be computed with:

$$\begin{cases} \text{Error} = \min_{i=1}^{n}\{\sqrt{(x_c - x_{\text{gt}_i})^2 + (y_c - y_{\text{gt}_i})^2}\} \\ \text{Var}(x_{gt} + y_{gt}) = Var(x_{gt}) + Var(y_{gt}) + 2 \cdot COV(x_{gt}, y_{gt}) \end{cases} \tag{5.1}$$

where (x_c, y_c) is the estimated point by GPT-4 and $(x_{\text{gt}_i}, y_{\text{gt}_i})$ are the ground truth points computed by the user. A point is considered to be the ground truth if $f_{\text{gt}_i} > 0.8$.

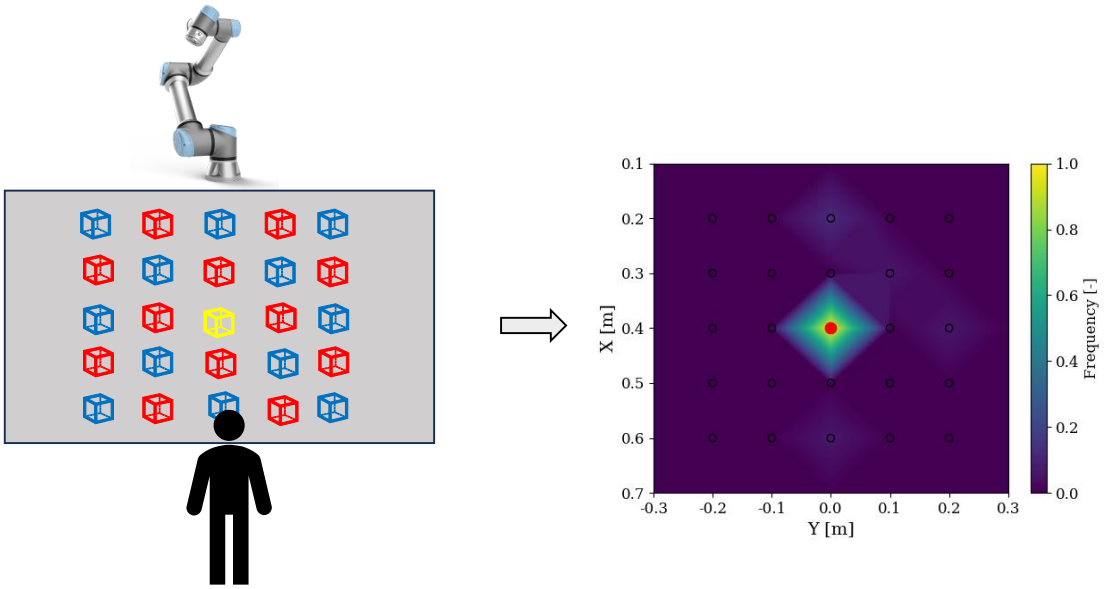


Figure 5.12: (Left) The representation of the layout with the yellow cube representing the cube chosen by the robot to be picked up. (Right) The graphic representation with the answers from GPT-4 and the results from the user study. In red is the answer from GPT-4.

Figure 5.13 compares the response from GPT-4 (in red) and the responses from the user study. On the left side, the prompt provided to both the user and GPT-4 is "*Take a block that is close to me.*" On the right side, the prompt is "*Take a block that is close to the robot.*" GPT-4 and the users agree on which block the user should pick in both scenarios. By utilizing Equation 5.1, the error for this case is calculated to be zero. This implies that GPT-4 is capable of suggesting 3D poses with respect to both the robot and the user.

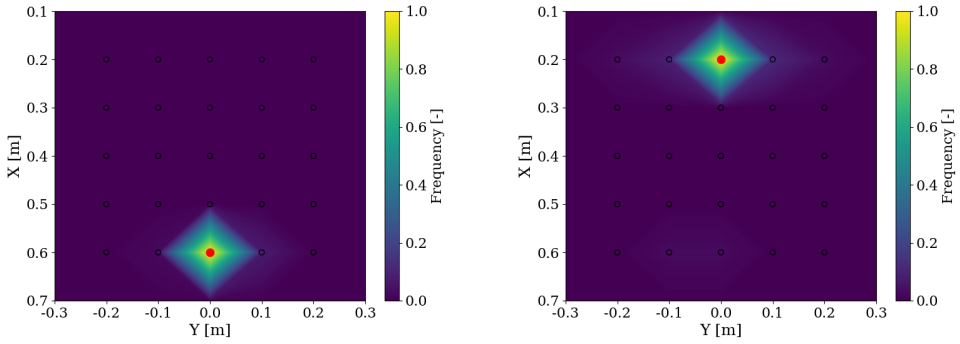


Figure 5.13: Comparison between the user study results and GPT-4 answers (in red) with different prompts: (Left) "*Take a block that is close to me.*" (Right) "*Take a block that is close to the robot.*"

In contrast to the previous case where the prompt resulted in minimal variance in the user's response, the prompts used in Figure 5.14 demonstrate otherwise. On the left side, the prompt is "*Take a block on the left,*" and on the right side, the prompt is "*Take a cube on the right.*" Such prompts do not lead to a unique solution. Furthermore, the prompt was slightly modified to investigate the impact of changing the objective terminology. Instead of referring to the objective as "*a block,*", it was termed "*a cube*" for the prompt on the right. Despite this modification, no significant changes are observed in either GPT- 4's or the users' responses. This suggests that altering the terminology of the objective does not significantly affect the responses.

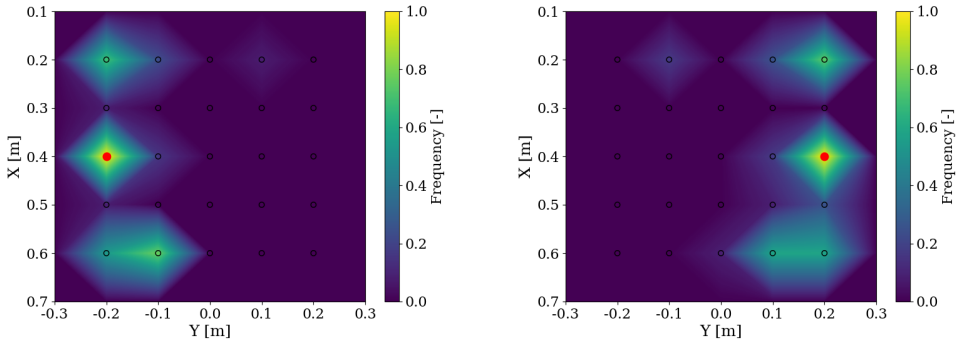


Figure 5.14: Comparison between the user study results and GPT-4 answers (in red) with different prompts: (Left) "*Take a block on the left.*" (Right) "*Take a cube on the right.*"

Similarly to the previous case where there was high variance, the prompt "*Take*

a block far from the robot" exhibits a non-deterministic response from the user. The prompt on the right side of Figure 5.15 demonstrates a clear division in the user's response. The prompt "*Take one of the furthest block from the robot,*" which does not specify a precise position, explains why the user responses are not deterministic. This lack of specificity in the prompt allows for interpretation and choice among multiple possible blocks that could be considered "*farthest from the robot.*"

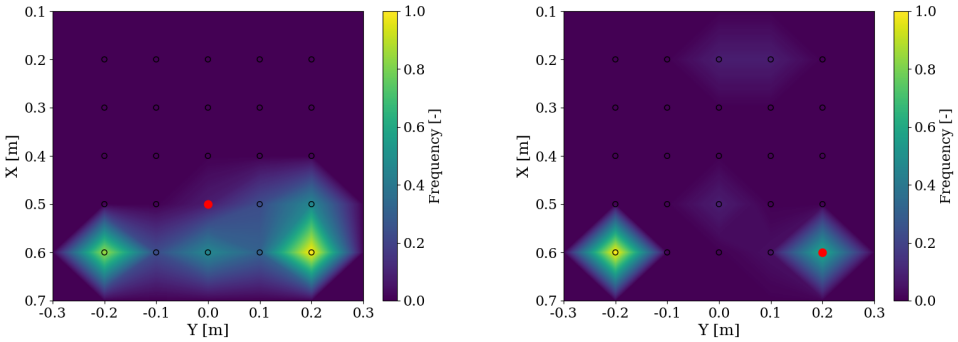


Figure 5.15: Comparison between the user study results and GPT-4 answers (in red) with different prompts: (Left) "*Take a block far from the robot.*" (Right) "*Take one of the furthest block from the robot.*"

Table 5.9 presents each of the prompts used in the user study, along with the Error and Variance computed using Equation 5.1 for each prompt. Based on the definition used to assess the error produced by GPT-4, the proposed method usually tends to find the correct solution. Specifically, more than 50% of the solutions match exactly where the user intended the robot to go (according to Equation 5.1). As discussed previously, the maximum error occurs in cases where prompts are more subjective, but even then, it remains minimal. The maximum error is 0.4 m, which happens when the prompt "*Take one of the furthest block from the robot*" is used. The users' answers are mostly distributed among two points, but because the point on the left was chosen more often, the point chosen by GPT-4 was not deemed the correct one. One can imagine that the error could be reduced to 0 m with enough data points.

Prompt	**Error** [m]	**Var** [m^2]
Take a block on the left	**0.0**	0.027
Take a cube on the right	**0.0**	0.030
Take a block on the right and close to me	0.1	0.005
Take a block close to me	**0.0**	**0.001**
Take a block far from me	0.1	0.033
Take a block between me and the robot	**0.0**	0.004
Take a block close to the robot	**0.0**	0.012
Take a block far from the robot	0.22	0.032
Take one of the furthest block from the robot	**0.4**	**0.052**

Table 5.9: Error and Var computed with Equation 5.1 for the prompts used in the user study. A lower error means that the answer from GPT-4 and the users are closer. Higher variance means that the answers from the users are more spread.

Chapter 6

Discussion

This section provides a discussion of key results from the different experiments. Initially, it discusses the results derived from quantitative and qualitative experiments to address the hypothesis concerning safety perception. Subsequently, it evaluates the proposed streamlined coding paradigm's balanced accuracy and emergent properties. Lastly, it analyses the qualitative results of the 3D poses proposed by GPT-4, exploring their appreciation for guiding trajectories.

6.1 Perception of Safety

In the initial phase of the results analysis, the GPT_f estimated by users were compared to those suggested by GPT-4. It was found that GPT-4 demonstrated a high level of accuracy in predicting factors that reflect users' states of mind, as evidenced by the close alignment between its outputs and those provided by users. However, considering the vast array of natural language inputs available, it is essential to acknowledge the limited scope of prompts tested in this comparison. To better quantify the relationship between GPT-4 and human responses, extending the experiment and incorporating a more comprehensive range of prompts is imperative.

Furthermore, using these factors in conjunction with an existing algorithm to adapt the safety evaluation of a robot based on users' feelings was evaluated, employing both qualitative and quantitative assessments. The robot's behavior yielded expected results, particularly in situations considered challenging. The task became unfeasible when users exhibited a heightened fear of robots, which is a desirable outcome. For instance, if a user demonstrated significant apprehension, the robot refrained from tasks involving close proximity, such as placing objects directly into their hands. Contrarily, when users displayed confidence, the robot resumed its activities, demonstrating the effective adaptation of robot behavior

based on user sentiment. Additionally, adapting its average speed and average distance to the user suggests that the robot can interpret the user's feelings. To further evaluate the method, feedback from users to ascertain their satisfaction with the observed changes in robot behavior was used. Positive responses validate not only the efficacy of behavior modifications but also the fact that LLMs can indeed incorporate human perceptions of safety into safety evaluations.

However, it is crucial to highlight the specific characteristics of the experiments, particularly in the context of the final assessment. Users who provided feedback on changes in robot behavior did not directly interact with the robot, potentially impacting their responses compared to those who would. Additionally, it is noteworthy that all users who participated in the experiment were not asked about their stance towards robots prior to the experiment, which could potentially bias the results. To further validate the system, it would be interesting to find subjects who are afraid of robots and assess whether the method actually meets the user's expectations.

6.2 Streamlined Coding Paradigm

In the initial part of the results, where the three factors are tested, limitations of the method became apparent. While paraphrasing effectively handles simple conditions, GPT-4 struggles to maintain acceptable balanced accuracy as conditions grow more complex. This underscores the inherent limitations of GPT-4, which is best suited for straightforward conditions with minimal information to consider. Although it might be feasible to address this limitation by segmenting the problem into subproblems, achieving higher accuracy for more complex conditions remains challenging.

The subsequent part, focusing on emergent properties, also presents noteworthy findings. Despite the absence of a singular correct answer, the robot adeptly continues working in a manner that aligns with the context of all explored cases. However, it is essential to acknowledge the potential drawbacks of this capability. A robot that persists in its tasks despite encountering contradictions could inadvertently reinforce user errors, leading to potential issues.

These results suggest caution in employing such systems in critical scenarios where erroneous outcomes carry significant consequences, such as in production lines or surgical robots. However, in scenarios where the impact of errors is minimal, such as in specific experimental or non-critical settings, the system could still prove useful, particularly when considering user satisfaction and frustration levels as additional metrics. Despite its limitations, it remains a valuable tool in contexts where even imperfect assistance is preferable to none.

6.3 Trajectory Shaping through 3D Poses

The results from the experiment in which the robot must change its objective showcase the robot's ability to suggest 3D poses accurately. Indeed, the robot consistently selected the same cube as the users in over 50% of trials. Even in instances where the robot's choice differed from the user's, its selections remained remarkably close. However, exploring how varying the number of cubes influences GPT-4's responses would be valuable. As observed in the streamlined coding paradigm, increasing input conditions tend to reduce GPT-4's performance. A similar trend is likely to emerge in this scenario, indicating the model's limitations under more complex input conditions.

In the analysis of the withdrawal experiment, it is evident that the parking attraction force in withdrawal 2 may have been marginally insufficient, given the similarity in results to the withdrawal without any user impact. Nevertheless, there was a slight improvement, indicating the method's effectiveness, although with room for enhancement. In contrast, withdrawal 3, where GPT-4 directly suggested the withdrawal position, yielded more satisfactory outcomes. Moreover, the algorithmic governance of the safety aspect in the suggested 3D poses presents an intriguing aspect. Given the potential for language models to generate erroneous outputs, this approach offers a balance between safety considerations and user interaction. Interestingly, user feedback highlighted the paramount importance of robot compliance over human-like behavior. This aligns with the task's objective of efficient and intelligent collaboration between the robot and the user rather than simulating human characteristics. Furthermore, the results remained consistent regardless of the participant's background, indicating a lack of distinction based on cultural or robotic experience.

These results show that LLMs can be used to suggest 3D poses that shape trajectories with zero-shot prompting. One should be cautious, however, since the robot can sometimes hallucinate and suggest poses that are out of the workspace or do not follow directives provided by the user. As it was explored in this thesis, algorithmic governance might be the right trade-off between safety and user desires. More work could be done to evaluate the potential of such methods further.

Chapter 7

Conclusion

This work proposes a framework aimed at enhancing collaboration between robots and humans. Leveraging Large Language Models (LLMs) and Visual Language Models (VLMs), the framework facilitates adaptive safety evaluations based on users' safety perceptions, enabling the robot's behavior to adapt accordingly and ensure a safe environment for all users. When faced with potentially hazardous situations, the robot employs a streamlined coding paradigm to autonomously determine its next course of action, thus facilitating seamless collaboration between robots and humans. This addresses the demanding challenge of natural language guidance of robot behavior. Additionally, the system shapes robotic arm trajectories based on user preferences by suggesting 3D poses.

Experiments were designed to evaluate the effectiveness and robustness of the proposed framework by investigating three distinct hypotheses presented below. Firstly, a user-base study compared users' answers with answers from GPT-4 to estimate human safety perceptions through a scaling factor. The same users evaluated their satisfaction regarding the robot's behavior changes when provided with different perceptions of safety. A quantitative evaluation of the time required to finish collaborative tasks and the success rate was also done. Secondly, an assessment of the utilization of LLMs for decision-making in HRC by providing a streamlined coding paradigm was performed. The paraphrasing of conditions, adjustment, and scalability were assessed first. Then, emergent properties of LLMs were leveraged to see how the system would react in the case of contradictory or ambiguous prompts from the user. Finally, another user study evaluated the withdrawal and change of objective methods based on users' satisfaction.

The results show that the scaling factor representing the perception of safety estimated by GPT-4 closely resembles the answers provided by the users, with an RMSE of 0.144. The robot's behavior effectively changes with the different scaling factors, such as the completion time, average distance between the user and robot end-effector, and end-effector speed adapt to the safety index. Additionally, users

are generally satisfied with these changes, with a median satisfaction level above 3 (on a scale of 1 to 5 with 5 being the most satisfied). However, the variance in the streamlined coding paradigm increases with the complexity of the input conditions, resulting in balanced accuracy below 50% in some cases. Even with simpler conditions and paraphrasing, the balanced accuracy does not reach 100%, unlike conventional coding approaches. Finally, the withdrawal method where GPT-4 directly suggests the withdrawal positions proved to be more satisfying for users than methods not influenced by GPT-4, with 70% favoring the method where GPT-4 directly suggests the withdrawal position and 12% without any influence from GPT-4. When GPT-4 was used to suggest new objective positions, the suggestions matched the users' answers more than 50% of the time.

The contributions of this research were evaluated through these carefully designed experiments to address the following hypotheses:

- LLMs can integrate human safety perceptions into safety evaluations: **accepted**, the results indicate a positive direction however further extensive research is warranted.
- LLMs can facilitate decision-making processes in Human-Robot Collaboration (HRC) through a streamlined coding paradigm: **neither**, the findings through the experiments realized remain inconclusive unless the application tolerates errors from the robot.
- LLMs can shape trajectories by suggesting 3D poses based on natural language input: **accepted**, the results affirm the feasibility of this approach, although implementing a constraint satisfaction algorithm is essential to address instances where the foundational model may produce erroneous outputs.

There are numerous avenues for further exploration. For instance, the experiments could be refined by increasing the number of prompts used to assess the accuracy of GPT_f estimation. Similarly, safety experiments should involve in-person interactions with pre-selected participants who express varying confidence levels around robots. Addressing scalability issues in the streamlined coding paradigm could involve systematically dividing conditions into subproblems. Furthermore, in the trajectory shaping experiment, adjusting the attraction force of the hybrid withdrawal method could enhance performance.

Two notable works relevant to this thesis were recently published during IEEE ICRA 2024[1]. The first paper by [9] introduces a world model that creates a surrogate environment for training controllers and predicting safety violations by learning the internal dynamic model of systems. This approach leverages foundational world

[1]ICRA2024, held from May 13th to May 16th, 2024: `https://2024.ieee-icra.org`

models to further reduce dataset requirements. The second paper by [74] presents a methodology for quantifying the complex relationship between perception metrics and robotic metrics, a relationship that remains largely unclear. These works could significantly enhance the comparative analysis of safety metrics and provide a rigorous quantitative assessment of both perception-based and robotic safety evaluations.

In a broader context, it would be interesting to explore the versatility of this framework. For example, extending the variety of objects on the table and utilizing the decision-making process for diverse purposes, such as object selection and manipulation, could be explored. Additionally, incorporating mixed reality lenses into the framework could introduce new dimensions to the collaboration between robots and humans, further enhancing interaction and task execution. Finally, as the LLMs keep on improving, producing the same experiments with newer models could be interesting. GPT-4o[2] was recently released and offers real-time multimodal perception which could potentially improve the results obtained in this thesis.

[2]GPT-4o: https://openai.com/index/hello-gpt-4o/

Bibliography

[1] A. Grybauskas, *Industry 5.0 and Artificial Semi-General Intelligence. Exploring Future Challenges and Opportunities Within Industries and Societies*, 04 2024, pp. 93–112.

[2] J. Wang, E. Shi, S. Yu, Z. Wu, C. Ma, H. Dai, Q. Yang, Y. Kang, J. Wu, H. Hu, C. Yue, H. Zhang, Y. Liu, Y. Pan, Z. Liu, L. Sun, X. Li, B. Ge, X. Jiang, D. Zhu, Y. Yuan, D. Shen, T. Liu, and S. Zhang, "Prompt Engineering for Healthcare: Methodologies and Applications," arXiv preprint arXiv:2304.14670, 2024.

[3] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman, B. Ichter, D. Driess, J. Wu, C. Lu, and M. Schwager, "Foundation Models in Robotics: Applications, Challenges, and the Future," arXiv preprint arXiv:2312.07843, 2023.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," arXiv preprint arXiv:1706.03762, 2023.

[5] OpenAI, "GPT-4 Technical Report," arXiv preprint arXiv:2303.08774, 2023.

[6] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision," arXiv preprint arXiv:2103.00020, 2021.

[7] S. Fujii and Q.-C. Pham, "Time-Optimal Path Tracking with ISO Safety Guarantees," arXiv preprint arXiv:2306.05197, 2023.

[8] E. Coronado, T. Kiyokawa, G. A. Garcia Ricardez, I. G. Ramirez-Alpizar, G. Venture, and N. Yamanobe, "Evaluating quality in human-robot interaction: A systematic search and classification of performance and human-centered factors, measures and metrics towards an industry 5.0," *Journal of Manufacturing Systems*, vol. 63, pp. 392–410, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612522000577

[9] Z. Mao, S. Dai, Y. Geng, and I. Ruchkin, "Zero-shot Safety Prediction for Autonomous Robots with Foundation World Models," arXiv preprint arXiv:2404.00462, 2024.

[10] A. Zacharaki, I. Kostavelis, A. Gasteratos, and I. Dokas, "Safety bounds in human robot interaction: A survey," *Safety Science*, vol. 127, p. 104667, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753520300643

[11] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, S. Vemprala, and R. Bonatti, "LATTE: LAnguage Trajectory TransformEr," arXiv preprint arXiv:2208.02918, 2022.

[12] T. Taniguchi, S. Murata, M. Suzuki, D. Ognibene, P. Lanillos, E. Ugur, L. Jamone, T. Nakamura, A. Ciria, B. Lara, and G. Pezzulo, "World Models and Predictive Coding for Cognitive and Developmental Robotics: Frontiers and Challenges," arXiv preprint arXiv:2301.05832, 2023.

[13] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring Massive Multitask Language Understanding," arXiv preprint arXiv:2009.03300, 2021.

[14] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei, "Scaling Instruction-Finetuned Language Models," arXiv preprint arXiv:2210.11416, 2022.

[15] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "HellaSwag: Can a Machine Really Finish Your Sentence?" arXiv preprint arXiv:1905.07830, 2019.

[16] X. Liu, H. Cheng, P. He, W. Chen, Y. Wang, H. Poon, and J. Gao, "Adversarial Training for Large Neural Language Models," arXiv preprint arXiv:2004.08994, 2020.

[17] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H.

Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba, "Evaluating Large Language Models Trained on Code," arXiv preprint arXiv:2107.03374, 2021.

[18] B. Chen, F. Zhang, A. Nguyen, D. Zan, Z. Lin, J.-G. Lou, and W. Chen, "CodeT: Code Generation with Generated Tests," arXiv preprint arXiv:2207.10397, 2022.

[19] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, "DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs," arXiv preprint arXiv:1903.00161, 2019.

[20] K. Chen, W. Xu, X. Cheng, Z. Xiaochuan, Y. Zhang, L. Song, T. Wang, Y. Qi, and W. Chu, "Question Directed Graph Attention Network for Numerical Reasoning over Text," arXiv preprint arXiv:2009.07448, 2023.

[21] G. Team, M. Reid, and N. Savinov, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," arXiv preprint arXiv:2403.05530, 2024.

[22] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang, "Fast Segment Anything," arXiv preprint arXiv:2306.12156, 2023.

[23] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything," arXiv preprint arXiv:2304.02643, 2023.

[24] C. Zhang, D. Han, Y. Qiao, J. U. Kim, S.-H. Bae, S. Lee, and C. S. Hong, "Faster Segment Anything: Towards Lightweight SAM for Mobile Applications," arXiv preprint arXiv:2306.14289, 2023.

[25] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting Twenty-thousand Classes using Image-level Supervision," arXiv preprint arXiv:2201.02605, 2022.

[26] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," arXiv preprint arXiv:2212.04356, 2022.

[27] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," arXiv preprint arXiv:2006.11477, 2020.

[28] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units," 2021, arXiv preprint arXiv:2106.07447, 2021.

[29] C. Zheng, W. Wu, C. Chen, T. Yang, S. Zhu, J. Shen, N. Kehtarnavaz, and M. Shah, "Deep Learning-based Human Pose Estimation: A Survey," *ACM Comput. Surv.*, vol. 56, no. 1, aug 2023. [Online]. Available: https://doi.org/10.1145/3603618

[30] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device Real-time Body Pose tracking," arXiv preprint arXiv:2006.10204, 2020.

[31] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," arXiv preprint arXiv:1812.08008, 2019.

[32] S. Mroz, N. Baddour, C. McGuirk, P. Juneau, A. Tu, K. Cheung, and E. Lemaire, "Comparing the Quality of Human Pose Estimation with BlazePose or OpenPose," in *2021 4th International Conference on Bio-Engineering for Smart Technologies (BioSMART)*, 2021, pp. 1–4.

[33] T. Yu, Z. Zheng, K. Guo, J. Zhao, Q. Dai, H. Li, G. Pons-Moll, and Y. Liu, "DoubleFusion: Real-time Capture of Human Performances with Inner Body Shapes from a Single Depth Sensor," arXiv preprint arXiv:1804.06023, 2018.

[34] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," arXiv preprint arXiv:1612.00593, 2017.

[35] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," arXiv preprint arXiv:1706.02413, 2017.

[36] Y. Chen, J. Arkin, C. Dawson, Y. Zhang, N. Roy, and C. Fan, "AutoTAMP: Autoregressive Task and Motion Planning with LLMs as Translators and Checkers," arXiv preprint arXiv:2306.06531, 2024.

[37] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and Where Pathways for Robotic Manipulation," arXiv preprint arXiv:2109.12098, 2021.

[38] Adaptive Agent Team, J. Bauer, K. Baumli, S. Baveja, F. Behbahani, A. Bhoopchand, N. Bradley-Schmieg, M. Chang, N. Clay, A. Collister,

V. Dasagi, L. Gonzalez, K. Gregor, E. Hughes, S. Kashem, M. Loks-Thompson, H. Openshaw, J. Parker-Holder, S. Pathak, N. Perez-Nieves, N. Rakicevic, T. Rocktäschel, Y. Schroecker, J. Sygnowski, K. Tuyls, S. York, A. Zacherl, and L. Zhang, "Human-Timescale Adaptation in an Open-Ended Task Space," arXiv preprint arXiv:2301.07608, 2023.

[39] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3M: A Universal Visual Representation for Robot Manipulation," arXiv preprint arXiv:2203.12601, 2022.

[40] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as Policies: Language Model Programs for Embodied Control," arXiv preprint arXiv:2209.07753, 2023.

[41] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-BERT: Pre-training 3D Point Cloud Transformers with Masked Point Modeling," arXiv preprint arXiv:2111.14819, 2022.

[42] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, "CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields," arXiv preprint arXiv:2112.05139, 2022.

[43] Y. Ye, X. Li, A. Gupta, S. D. Mello, S. Birchfield, J. Song, S. Tulsiani, and S. Liu, "Affordance Diffusion: Synthesizing Hand-Object Interactions," arXiv preprint arXiv:2303.12538, 2023.

[44] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo, "EmbodiedGPT: Vision-Language Pre-Training via Embodied Chain of Thought," arXiv preprint arXiv:2305.15021, 2023.

[45] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019.

[46] T. Yoneda, J. Fang, P. Li, H. Zhang, T. Jiang, S. Lin, B. Picker, D. Yunis, H. Mei, and M. R. Walter, "Statler: State-Maintaining Language Models for Embodied Reasoning," arXiv preprint arXiv:2306.17840, 2023.

[47] Q. Lyu, J. Tan, M. E. Zapadka, J. Ponnatapura, C. Niu, K. J. Myers, G. Wang, and C. T. Whitlow, "Translating Radiology Reports into Plain Language using ChatGPT and GPT-4 with Prompt Learning: Promising Results, Limitations, and Potential," arXiv preprint arXiv:2303.09038, 2023.

[48] H. Dai, Z. Liu, W. Liao, X. Huang, Y. Cao, Z. Wu, L. Zhao, S. Xu, W. Liu, N. Liu, S. Li, D. Zhu, H. Cai, L. Sun, Q. Li, D. Shen, T. Liu, and X. Li,

"AugGPT: Leveraging ChatGPT for Text Data Augmentation," arXiv preprint arXiv:2302.13007, 2023.

[49] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, "A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications," arXiv preprint arXiv:2402.07927, 2024.

[50] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[51] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language Models as Knowledge Bases?" arXiv preprint arXiv:1909.01066, 2019.

[52] G. Marvin, N. Hellen, D. Jjingo, and J. Nakatumba-Nabende, "Prompt Engineering in Large Language Models," in *Data Intelligence and Cognitive Informatics*, I. J. Jacob, S. Piramuthu, and P. Falkowski-Gilski, Eds. Singapore: Springer Nature Singapore, 2024, pp. 387–402.

[53] L. Giray, "Prompt Engineering with ChatGPT: A Guide for Academic Writers," vol. 51, no. 12, pp. 2629–2633. [Online]. Available: https://doi.org/10.1007/s10439-023-03272-4

[54] K. Busch, A. Rochlitzer, D. Sola, and H. Leopold, "Just Tell Me: Prompt Engineering in Business Process Management," arXiv preprint arXiv:2304.07183, 2023.

[55] D. Trautmann, A. Petrova, and F. Schilder, "Legal Prompt Engineering for Multilingual Legal Judgement Prediction," arXiv preprint arXiv:2212.02199, 2022.

[56] Y. Jin, D. Li, Y. A, J. Shi, P. Hao, F. Sun, J. Zhang, and B. Fang, "RobotGPT: Robot Manipulation Learning from ChatGPT," arXiv preprint arXiv:2312.01421, 2023.

[57] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley, Z. Xu, D. Sadigh, A. Zeng, and A. Majumdar, "Robots That Ask For Help: Uncertainty Alignment for Large Language Model Planners," arXiv preprint arXiv:2307.01928, 2023.

[58] R. Luo, S. Zhao, J. Kuck, B. Ivanovic, S. Savarese, E. Schmerling, and M. Pavone, "Sample-efficient safety assurances using conformal prediction," *The International Journal of Robotics Research*, Dec. 2023. [Online]. Available: http://dx.doi.org/10.1177/02783649231221580

[59] G. A. Garcia Ricardez, A. Yamaguchi, J. Takamatsu, and T. Ogasawara, "Human Safety Index Based on Impact Severity and Human Behavior Estimation," in *Mechatronics and Robotics Engineering for Advanced and Intelligent Manufacturing*, D. Zhang and B. Wei, Eds. Cham: Springer International Publishing, 2017, pp. 177–190.

[60] N. M. Ikuta K, Ishii H, "Safety Evaluation Method of Design and Control for Human-Care Robots." *The International Journal of Robotics Research*, vol. 22, pp. 281–297, 2003.

[61] S. Haddadin, A. Albu-Schaffer, M. Frommberger, J. Rossmann, and G. Hirzinger, "The "DLR Crash Report": Towards a standard crash-testing protocol for robot safety - Part I: Results," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 272–279.

[62] C.-S. Tsai, J.-S. Hu, and M. Tomizuka, "Ensuring safety in human-robot coexistence environment," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 4191–4196.

[63] Haddadin, Sami and Albu-Schaffer, Alin and Frommberger, Mirko and Rossmann, Jurgen and Hirzinger, Gerd, "The "dlr crash report": Towards a standard crash-testing protocol for robot safety - part ii: Discussions," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 280–287.

[64] D. Kulic and E. Croft, "Pre-collision safety strategies for human-robot interaction," *Auton. Robots*, vol. 22, pp. 149–164, 01 2007.

[65] G. A. Garcia Ricardez, A. Yamaguchi, J. Takamatsu, and T. Ogasawara, "Asymmetric Velocity Moderation for human-safe robot control," *Advanced Robotics*, vol. 29, no. 17, pp. 1111–1125, 2015. [Online]. Available: https://doi.org/10.1080/01691864.2015.1034173

[66] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision Detection and Reaction: A Contribution to Safe Physical Human-Robot Interaction," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3356–3363.

[67] S. Fujii and Q.-C. Pham, "Time-Optimal Path Tracking with ISO Safety Guarantees," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2023. [Online]. Available: http://dx.doi.org/10.1109/IROS55552.2023.10342287

[68] M. Rubagotti, I. Tusseyeva, S. Baltabayeva, D. Summers, and A. Sandygulova, "Perceived safety in physical human–robot interaction—A survey," *Robotics and Autonomous Systems*, vol. 151, p. 104047, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889022000173

[69] T. Arai, R. Kato, and M. Fujita, "Assessment of operator stress induced by robot collaboration in assembly," *CIRP Annals*, vol. 59, no. 1, pp. 5–8, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0007850610000442

[70] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting Twenty-thousand Classes using Image-level Supervision," arXiv preprint arXiv:2201.02605, 2022.

[71] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "MediaPipe: A Framework for Perceiving and Processing Reality," in *Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR) 2019*, 2019. [Online]. Available: https://mixedreality.cs.cornell.edu/s/NewTitle_May1_MediaPipe_CVPR_CV4ARVR_Workshop_2019.pdf

[72] G. A. Garcia Ricardez, A. Yamaguchi, J. Takamatsu, and T. Ogasawara, "Withdrawal strategy for human safety based on a virtual force model," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1119–1124.

[73] L. El Hafi, G. A. Garcia Ricardez, F. von Drigalski, Y. Inoue, M. Yamamoto, and T. Yamamoto, "Software development environment for collaborative research workflow in robotic system integration," *Advanced Robotics*, vol. 36, no. 11, pp. 533–547, 2022. [Online]. Available: https://doi.org/10.1080/01691864.2022.2068353

[74] X. Zhang, J. Chong, and K. Youcef-Toumi, "How Does Perception Affect Safety: New Metrics and Strategy," arXiv preprint arXiv:2312.07744, 2024.

Appendices

8.1 Speech Recognition

You are a robot stack maker which has to interpret the message from a user and guess which action he is speaking about.

Note on the output process:
There are 4 possible actions that the user might be referring to: [1] The perception of safety of the user (with respect to the robot or just his feelings), [2] Condition on the input for the decision to make (decision is either wait, withdraw or change objective), [3] Information on the direction where the robot should withdraw (this withdrawal is where the robot should move to), [4] Information on the direction where the robot should pick up a block (which block he should pick and/or where). If you believe that the input has absolutely nothing to do with any of these answers, please send [5].

Special request from the user:
The message of the user is: *I am very confident around robots*

Output formatting:
Which action is this prompt describing? Your answer should be only the number of the action formatted as follows (with no other text): [x] where x is the number of the action.

Prompt 1: Speech recognition prompt used to decide on which topic to publish the text.

8.2 Safety Index Estimation

You are a collaborative robot that has as objective to pick up and stack all the blocks on the table. The robot (you) is working with a human collaborator on the same task and in the same workspace. The human collaborator is also stacking blocks with the robot. The human collaborator should give you a sentence that expresses how safe he feels around robots. Based on that sentence, you need to suggest a safety index that ranges from 0 to 2. 2 means that the robot is going to act in a very safe way by moving much slower than usual and 0 is going to make the robot move much quicker than usual.

Note on the output process:
The lower the safety index, the less safe the user will be. Similarly, the higher the safety index, the safer the user will be. If you feel like the user is not in his right state of mind, you should make sure he is protected and adjust the safety index accordingly. You should only give 0 or 2 in rare occasions, instead, you should try to stay in the range of 0.5 to 1.5. But of course, if the user expresses that he wants the maximum, you should give it to him. For example, Safety Index = 1.0 is a totally neutral use, someone who is confident could be safety index = 0.8 and someone who is not confident could be safety index = 1.2.

Special request from the user:
The way the user is feeling is: *Very confident around robots*

Output formatting:
What safety index would you suggest for this user? When you answer, only give me the new value in the format: [x] where x is the safety index and not any other text.

Prompt 2: GPT factor (GPT_f) prompt used to estimate the perception of safety.

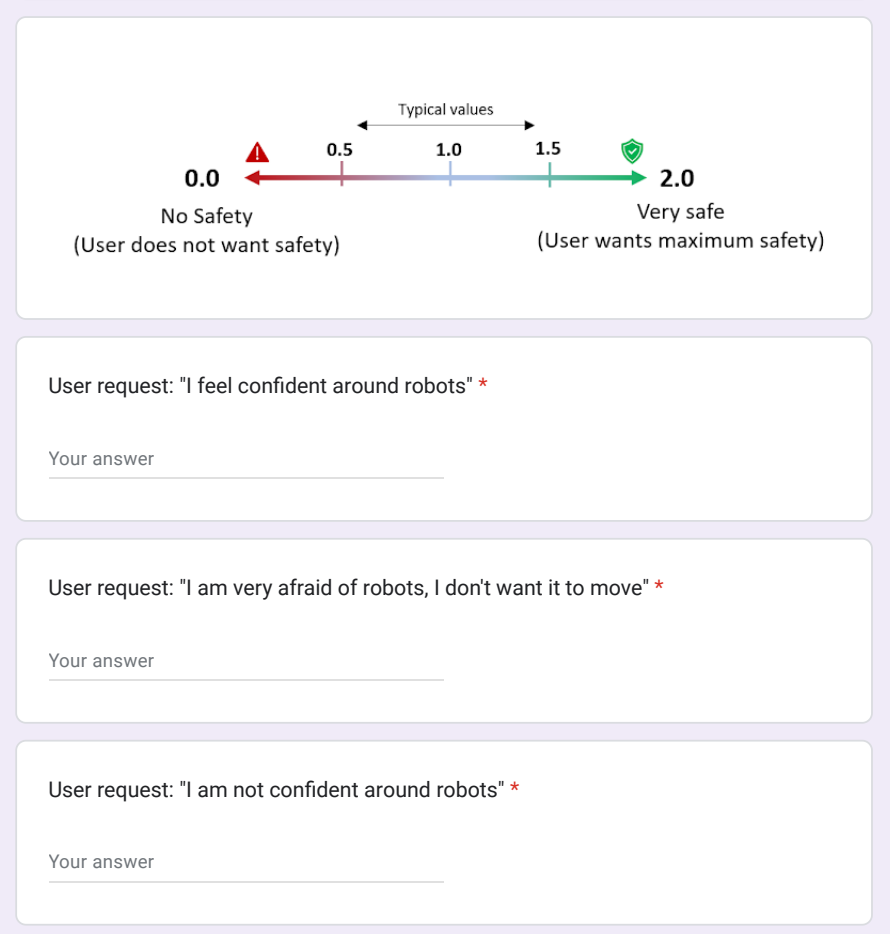


Figure 8.1: Survey preview for GPT_f estimation by the users: used for Experiment 4.2. Access to the whole experiment at: https://forms.gle/X7mmbZ1DL9G2DLQn8

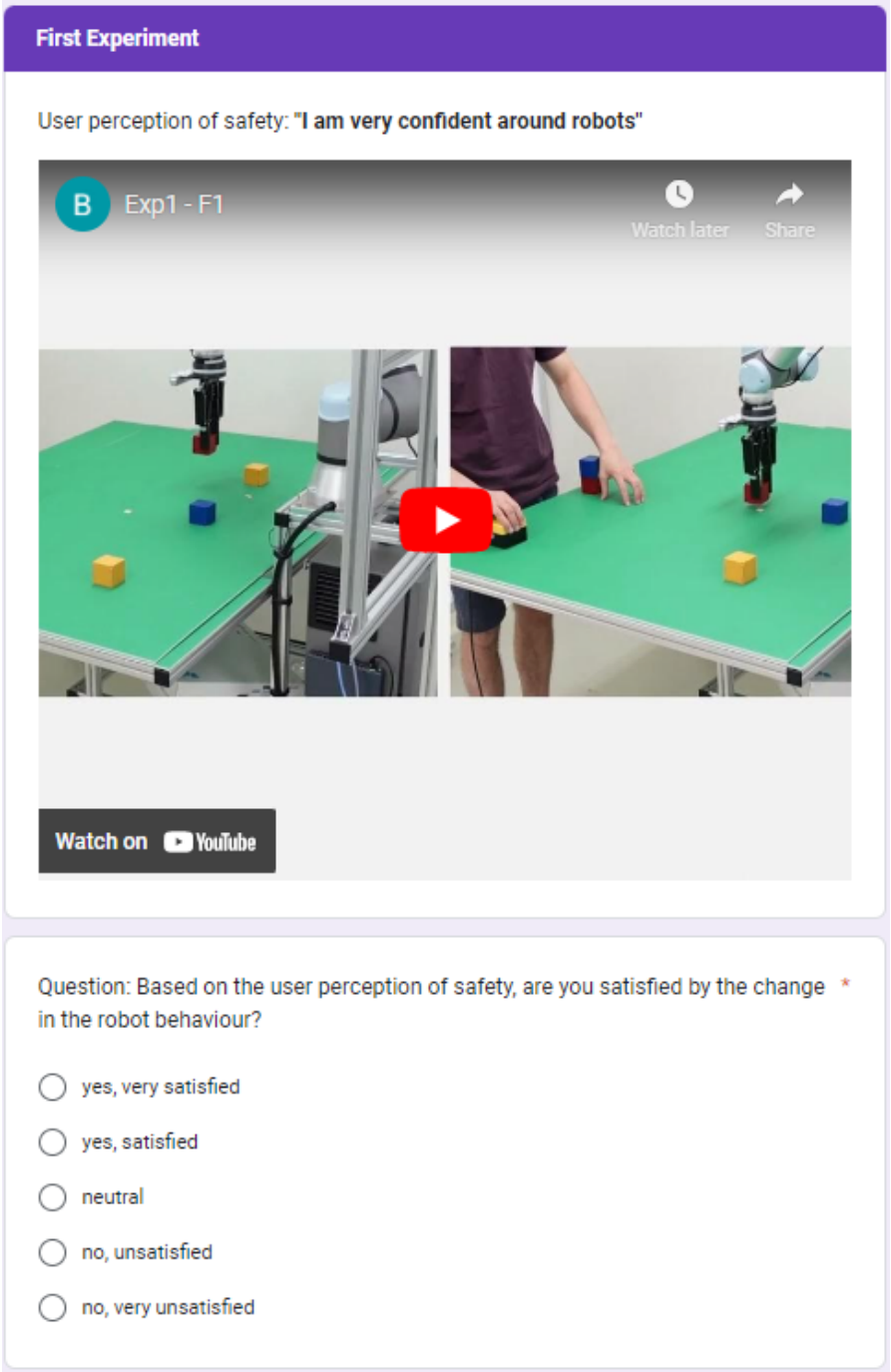
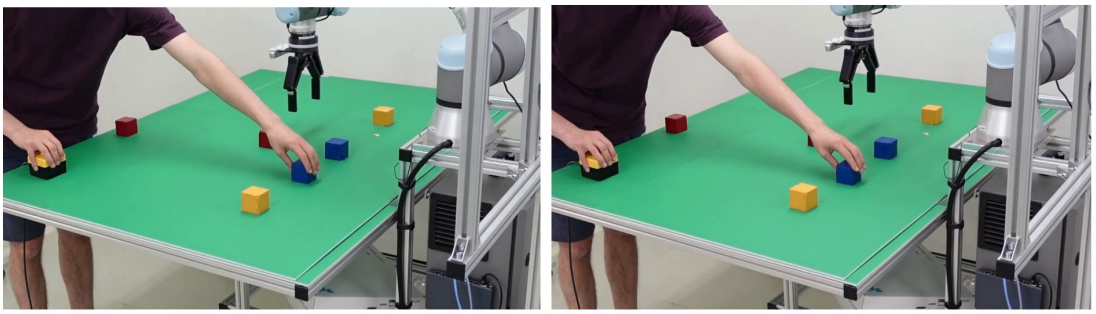
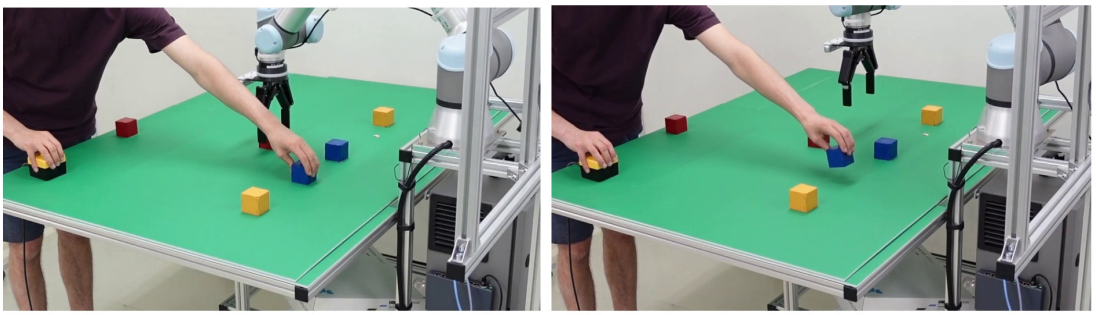


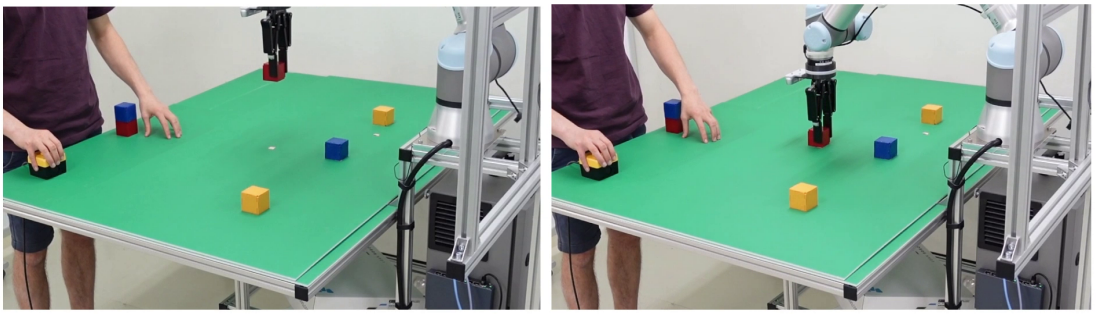
Figure 8.2: Survey preview for the satisfaction of the users for the change of behavior of the robot based on different GPT_f: used for Experiment 4.3. Access to the whole experiment at: https://forms.gle/zmfZVKKtUNSabntt5



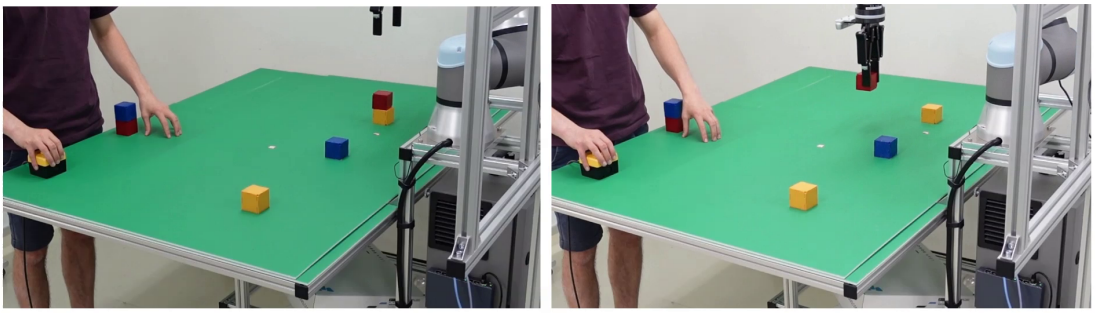
(a) Time elapsed = 1 s



(b) Time elapsed = 3 s



(c) Time elapsed = 7 s



(d) Time elapsed = 14 s

Figure 8.3: Images from the video used in a user study to compare $GPT_f = 0.5$ (left) and $GPT_f = 1.0$ (right) for the prompt "*I am very confident around robots*": used for the video showed in Figure 8.1. Full video available at: https://youtu.be/q9HaTDi29S0

8.3 Decision Making

You are a collaborative robot that has as objective to pick up and stack all the blocks on the table. The robot (you) is working with a human collaborator on the same task and in the same workspace. The human collaborator is also stacking blocks with the robot (sometimes on the same stack, sometimes not). You are going to get stuck because the situation is unsafe for the human or because there is an obstacle in your way. Based on three propositions, you need to decide what is the best possible action. When the user speaks about blocks, objects, or boxes, they all describe the same object. The end effector used to collect the boxes is a gripper.

The current state of the robot is:
The robot (you) is currently trying to pick up a block. The position of the current objective block is: *[x: -0.5m, y: 0.0m, z: 0.1m*]. The robot has been stuck in this position for: *2.0s*. There is currently *1* block left on the table ready to be picked up. The positions of the *1* objects are the following: *box1: [x: -0.45m, y: 0.05m, z: 0.1 m] dist = (0.07m)*, The distance between the end effector of the robot and the current objective is *0.1m*. The robot is stuck because it is in a dangerous situation which prevents him from moving closer to the objective. The position of the end-effector is *[x: -0.4m, y: 0.0m, z: 0.1 m]*.

Note on the decision-making process:
A situation that prompts safety concerns is usually due to a close distance between the human collaborator and the robot end-effector. An increase in the speed of the robot or the human can also prompt safety concerns. Waiting is advantageous if the distance with the current objective is rather small. Less than 0.2m is considered very small. If you have been stuck in a situation for more than 5 seconds, you should not wait anymore! Switching the objective is advantageous if the distance with other boxes is smaller than the current distance to the objective.

Special request from the user:
Always wait !

Decisions possible:
When you are placing, you should stick to your objective. You need to choose whether the robot should wait [1], go to an intermediate location, and try again to go to the objective [2] or try to go for another objective [3]. An intermediate location is computed via a withdrawal method which is considered to be a safe point. What is the best decision the robot could make? You need to choose one of the three functions. Your answer should be only the number of the function formatted as follows (with no other text): [x] where x is the number of the function.

Prompt 3: Decision-making prompt used to decide what the robot should do based on the user desire.

Prompts
Always change objective when there are 1 blocks left on the table ready to be picked up. Otherwise always wait . Never withdraw.
Always change objective when there are 0 blocks left on the table ready to be picked up. Otherwise always wait . Never withdraw.
Always change objective when there are 3 blocks left on the table ready to be picked up. Otherwise always wait . Never withdraw.
Always change objective when there are 1 blocks left on the table ready to be picked up. Otherwise always withdraw. Never wait .
Always change objective when there are 0 blocks left on the table ready to be picked up. Otherwise always withdraw. Never wait .
Always change objective when there are 3 blocks left on the table ready to be picked up. Otherwise always withdraw. Never wait .
Always wait when there are 1 blocks left on the table ready to be picked up. Otherwise always change objective . Never withdraw.
Always wait when there are 0 blocks left on the table ready to be picked up. Otherwise always change objective . Never withdraw.
Always wait when there are 3 blocks left on the table ready to be picked up. Otherwise always change objective . Never withdraw.
Always wait when there are 1 blocks left on the table ready to be picked up. Otherwise always withdraw. Never change objective .
Always wait when there are 0 blocks left on the table ready to be picked up. Otherwise always withdraw. Never change objective .
Always wait when there are 3 blocks left on the table ready to be picked up. Otherwise always withdraw. Never change objective .
Always withdraw when there are 1 blocks left on the table ready to be picked up. Otherwise always change objective . Never wait .
Always withdraw when there are 0 blocks left on the table ready to be picked up. Otherwise always change objective . Never wait .
Always withdraw when there are 3 blocks left on the table ready to be picked up. Otherwise always change objective . Never wait .
Always withdraw when there are 1 blocks left on the table ready to be picked up. Otherwise always wait . Never change objective .
Always withdraw when there are 0 blocks left on the table ready to be picked up. Otherwise always wait . Never change objective .
Always withdraw when there are 3 blocks left on the table ready to be picked up. Otherwise always wait . Never change objective .
Always change objective when picking. Otherwise always wait , Never withdraw.
Always change objective when picking. Otherwise always withdraw, Never wait .

Prompts
Always wait when picking. Otherwise always change objective , Never withdraw.
Always wait when picking. Otherwise always withdraw, Never change objective .
Always withdraw when picking. Otherwise always wait , Never change objective .
Always withdraw when picking. Otherwise always change objective , Never wait .
Always change objective when placing. Otherwise always wait , Never withdraw.
Always change objective when placing. Otherwise always withdraw, Never wait .
Always wait when placing. Otherwise always change objective , Never withdraw.
Always wait when placing. Otherwise always withdraw, Never change objective .
Always withdraw when placing. Otherwise always change objective , Never wait .
Always withdraw when placing. Otherwise always wait , Never change objective .
Always change objective when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always wait . Never withdraw.
Always change objective when the distance between the end effector of the robot and the current objective is $0.2\text{m} < \text{x} < 0.4\text{m}$. Otherwise always wait . Never withdraw.
Always change objective when the distance between the end effector of the robot and the current objective is > 0.4m. Otherwise always wait . Never withdraw.
Always change objective when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always withdraw. Never wait .
Always change objective when the distance between the end effector of the robot and the current objective is $0.2\text{m} < \text{x} < 0.4\text{m}$. Otherwise always withdraw. Never wait .
Always change objective when the distance between the end effector of the robot and the current objective is > 0.4m. Otherwise always withdraw. Never wait .
Always wait when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always change objective . Never withdraw.

Prompts
Always wait when the distance between the end effector of the robot and the current objective is 0.2m < x < 0.4m. Otherwise always change objective. Never withdraw.
Always wait when the distance between the end effector of the robot and the current objective is > 0.4m. Otherwise always change objective. Never withdraw.
Always wait when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always withdraw. Never change objective .
Always wait when the distance between the end effector of the robot and the current objective is 0.2m < x < 0.4m. Otherwise always withdraw. Never change objective .
Always wait when the distance between the end effector of the robot and the current objective is > 0.4m. Otherwise always withdraw. Never change objective .
Always withdraw when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always change objective . Never wait .
Always withdraw when the distance between the end effector of the robot and the current objective is 0.2m < x < 0.4m. Otherwise always change objective . Never wait .
Always withdraw when the distance between the end effector of the robot and the current objective is > 0.4m. Otherwise always change objective . Never wait .
Always withdraw when the distance between the end effector of the robot and the current objective is < 0.2m. Otherwise always wait . Never change objective .
Always withdraw when the distance between the end effector of the robot and the current objective is 0.2m < x < 0.4m. Otherwise always wait . Never change objective .
Always withdraw when the distance between the end effector of the robot and the current objective is > 0.4 m. Otherwise always wait . Never change objective .

Table 8.1: Prompts for condition adjustment, the prompts for condition paraphrasing and condition scaling are available at: https://github.com/Axtiop/GPTAlly_codebase.git

8.4 Robot Controller

8.4.1 Withdrawal

Algorithm 5 Withdrawal suggested by [72]

Require: $\Delta t, q_{\text{curr}}, d$
Ensure: q^*
1: Save current configuration for **PlaceBack**: $q_{\text{eng}} \leftarrow q_{\text{curr}}$
2: $F \leftarrow F_{\text{human}} + F_{\text{parking}}$
3: Obtain ΔV from F: $\Delta V \leftarrow \frac{F}{m}\Delta t$
4: Calculate the new velocity of the end-effector: $V' = V + \Delta V$
5: Calculate $\dot{q}'$ for V': $\dot{q}' \leftarrow J^{\#}(q_{\text{curr}})V'$
6: Obtain the target joint angles: $q(t + \Delta t) \leftarrow \dot{q}'\Delta t + q(t)$
7: AVM restricts the target joint angles: $q^* \leftarrow q(t + s\Delta t)$
8: Finish **TakeOut** and, thus, disengage **Withdrawal** if:
9: **if** $\|d\| > D_{\text{diseng}}$ **or** end-effector displacement $> D_{\text{withdrawing}}$ **or** end-effector arrives to the parking position **then**
10: **Finish TakeOut**
11: **end if**

You are a robot path planner who has to suggest a new position that the robot is going to be attracted to based on a language prompt. The reason you are needed is because there is currently an obstacle in the trajectory that doesn't allow the robot to continue its current trajectory. This position is called a parking position and it is considered a safe position.

Note on the decision making process:
The current position of the attraction point is: *[x: 0.3m, y: 0.0m, z: 0.4m]*. The user is located in [x: 0.7m, y: 0.0m, z: 0.1m]. This means that if he asks the point to be closer to him, x, y, and z should be closer to the user location. On the other hand, if he asks to go further away, the distance should increase. The distance is computed as sqrt(x^2 + y^2 + z^2). The robot's base is located in front of the user in position: [x: 0.0m, y: 0.0m, z: 0.0m]. The right side of the user is aligned with an increasing y. His other side, left, is aligned with a decreasing y. If the user asks to move more to the left, y should decrease. If the user asks to move more to the right, y should increase. Similarly, higher means z bigger, and lower means z smaller. It is important that the point that you suggest doesn't move the robot too close to the ground. The ground means z <= 0. There is a stack located in: [x: 0.3m, y:0.3, z: 0.1m]. Similarly to what was explained before, the user might ask about the position relative to the position of the stack. You must take the position of the stack into account..

Special request from the user:
The special request for this prompt that you must follow in addition to the angle requirement: *Move toward the left side of the stack!*

Output formatting:
What in your opinion is the best intermediate point for the robot? Your answer should be only your final proposition of the position formatted as follows (with no other text): [x: XXX, y: XXX, z: XXX] where XXX are the actual value you need to give in meters.

Prompt 4: Prompt provided to GPT-4 for typical withdrawal state for the mixed method.

You are a robot path planner who has to suggest an intermediate position (withdrawal position) where the robot should go before going to the initial goal position. The reason you are needed is because there is currently an obstacle in the trajectory that doesn't allow the robot to continue its current trajectory. However, you don't know the size or the shape. We suppose that the robot is moving in a linear trajectory from the current point to the objective.

The current state of the robot is:
The current obstacle direction is: *[x: 0.1m, y: 0.1m, z: 0.3m]*. The position of the end-effector is: *[x: 0.1m, y: 0.1m, z: 0.3m]*. The position of the obstacle is: *[x: 0.1m, y: 0.1m, z: 0.3m]*. The distance between the end-effector and the obstacle is: *0.2m*.

Note on the angle and ground restriction:
Just a reminder, the formula to compute the angle between two 3D vectors is cos^-1((a dot b)/(norm(a)*norm(b))). It is important that when you suggest a new position, the angle between the direction vector of this new trajectory position (which is computed as the new position goal - current position of the end-effector) and the current obstacle direction vector you propose should be bigger than 1.57 rad. In other words, it should move in the opposite direction. You must always check your suggestion with this condition and if it doesn't meet the requirement, you should suggest another position that matches this requirement. It is important that the point that you propose doesn't move the robot too close to the ground as it might be stuck between the obstacle and the ground. This can be verified with the value in the z axis which must be bigger than 0.2m.

Note on the user and object locations:
The user is located in [x: 0.7m, y: 0.0m, z: 0.1m]. This means that if he asks the point to be closer to him, the distance between him and the obstacle should decrease. On the other hand, if he asks to go further away, the distance should increase. The distance is computed as sqrt(x^2 + y^2 + z^2). The robot's base is located in front of the user in position: [x: 0.0m, y: 0.0m, z: 0.0m]. The right side of the user is aligned with an increasing y (right = y increase). His other side, left, is aligned with a decreasing y (left = y decrease). Similarly, higher means z bigger, and lower means z smaller. There is a stack located in: [x: 0.3m, y: 0.3m, z: 0.1m]. Similarly to what was explained before, the user might ask about the position relative to the position of the stack. You must take the position of the stack into account.

Special request from the user:
The special request for this prompt that you must follow in addition to the angle requirement: *Move toward the left side of the stack!*

Output formatting:
What in your opinion is the best intermediate point for the robot? Your answer should be only your final proposition of the position formatted as follows (with no other text): [x: X, y: X, z: X] where X are the actual value you need to give in meters.

Prompt 5: Prompt provided to GPT-4 for typical withdrawal state for the independent method.

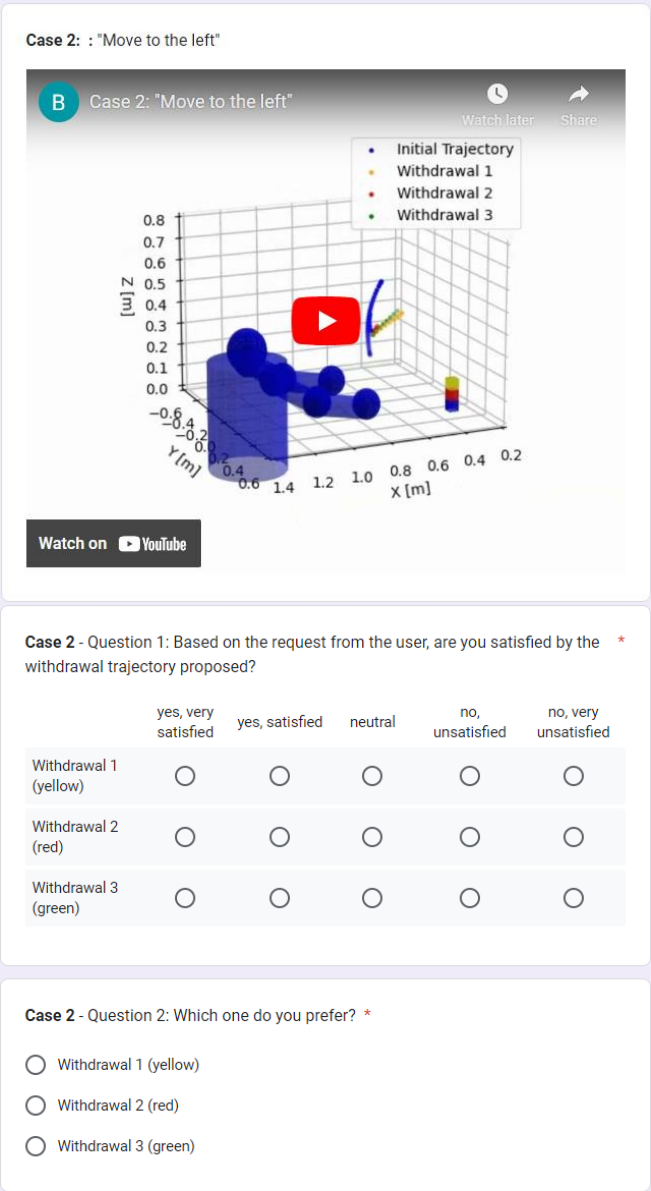


Figure 8.4: Survey preview for evaluating the satisfaction level of the users with the withdrawal methods: used for Experiment 4.5. Access to the whole experiment at: https://forms.gle/c7ufyJTc3WYbhCc59

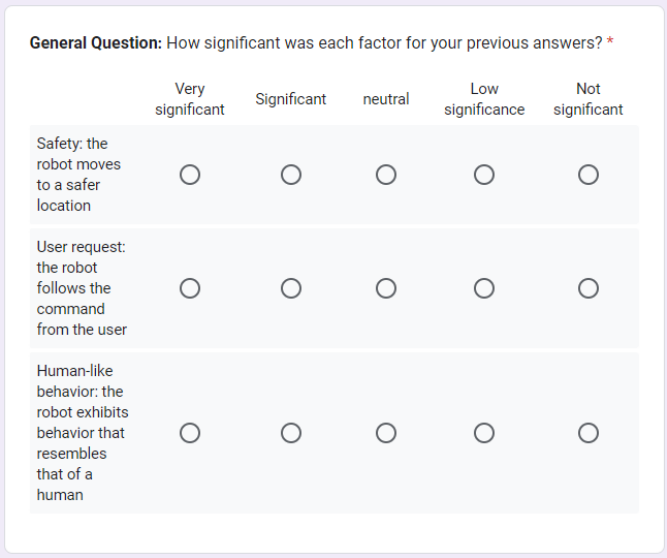
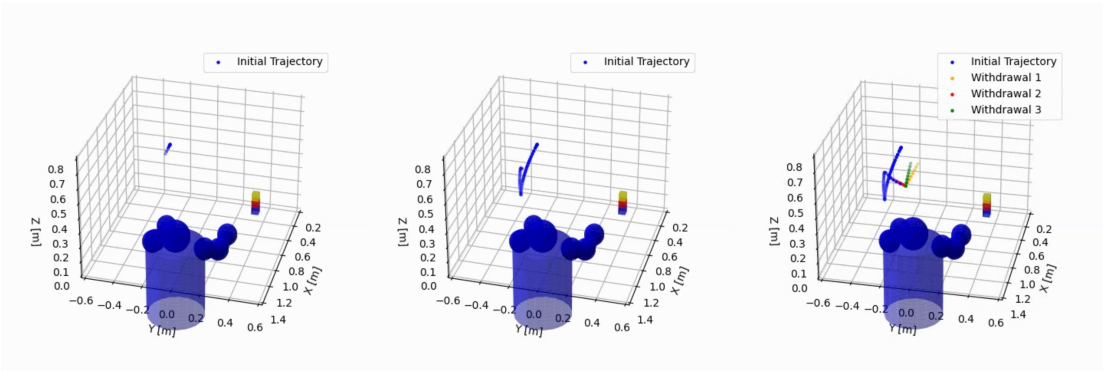


Figure 8.5: Survey preview for evaluating the importance of each factor influencing the decision made for the survey previewed in Figure 8.4: used for Experiment 4.5.



(a) Capture 1 (b) Capture 2 (c) Capture 3

(d) Capture 4 (e) Capture 5 (f) Capture 6

(g) Capture 7 (h) Capture 8 (i) Capture 9

Figure 8.6: Images from the video used in a user study to compare the three withdrawal methods for the prompt "*Move to the left*": used for the video showed in Figure 8.4. Full video available at: https://youtu.be/SaJbTiaCaJk

8.4.2 Changing Objective

You are a robot stack maker who has to suggest which cube to collect next. You are going to be given a natural language input describing the desire of the user regarding which cube you should be picking next. You will also be provided with the location of each cube. When the user speaks about blocks, objects, cubes, or boxes, they all describe the same type of objects.

The current state of the robot is:
The list with the position of each box and the distance between each box and the current position of the end-effector of the robot is *box1: [x: -0.45m, y: 0.05m, z: 0.1 m] dist = (0.07m), box2: [x: -0.45m, y: 0.05m, z: 0.1 m] dist = (0.07m), box3: [x: -0.45m, y: 0.05m, z: 0.1 m] dist = (0.07m)*,
Note on the position of the user and objects:
The user is located in *[x: 0.7m, y: 0.0m, z: 0.1m]*. This means that if he asks to pick up stacks closer to him, the x and y of the block should be closer to the user location (x increase and y closer to zero). On the other hand, if he asks to go further away, the distance between the user and the block should increase (x decreases and y further from zero). The robot's base is located in front of the user in position: [x: 0.0m, y: 0.0m, z: 0.0m]. The right side of the user is aligned with an increasing y. His other side, left, is aligned with a decreasing y. If the user asks to move more to the left, y should decrease. If the user asks to move more to the right, y should increase.

Special request from the user:
The request of the user that you must apply for your decision is: *Take a block between the robot and me!*

Output formatting:
Which object (box) should the robot pick up? Your answer should be only the number of the box formatted as follows (with no other text): [x] where x is the number of the box.

Prompt 6: Prompt provided to GPT-4 for typical change of objective state.

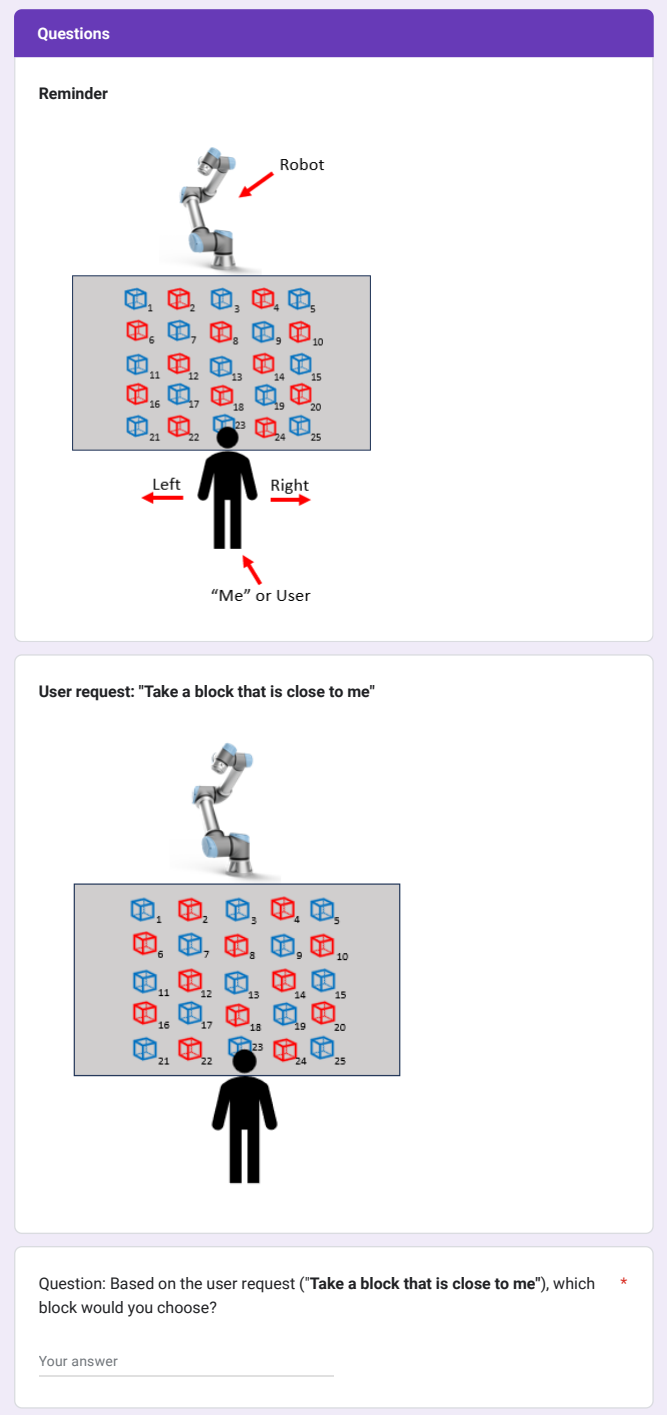
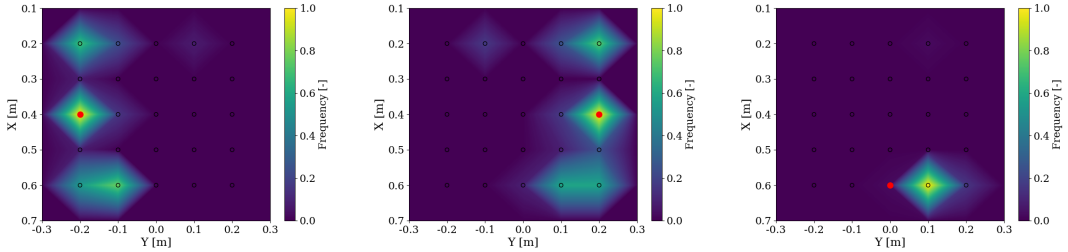


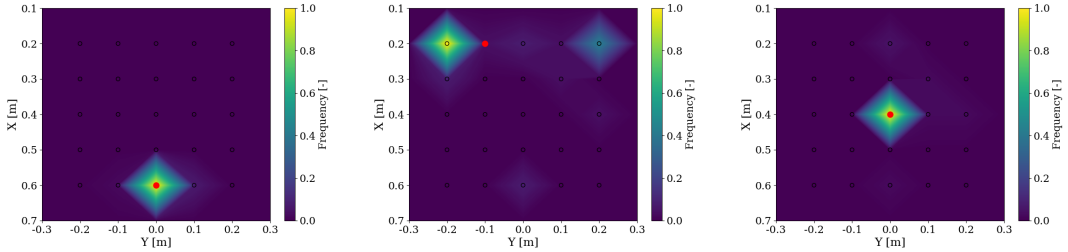
Figure 8.7: Survey preview for evaluating the accuracy of the positions suggested by GPT-4 through a user study: used for Experiment 4.5. Access to the whole experiment at: https://forms.gle/8uvvfsXHNpywFYQX9



(a) Pick a block on the left

(b) Pick a cube on the right

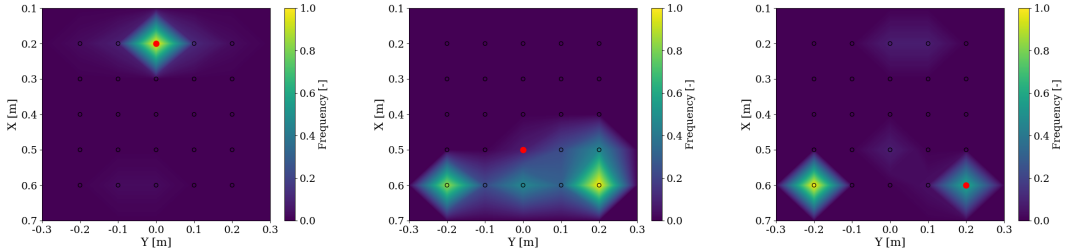
(c) Pick a block on the right and close to me



(d) Pick a block that is close to me

(e) Pick a block that is far from me

(f) Pick a block between me and the robot



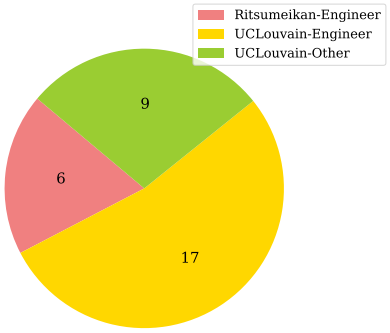
(g) Pick a block close to the robot

(h) Pick a block far from the robot

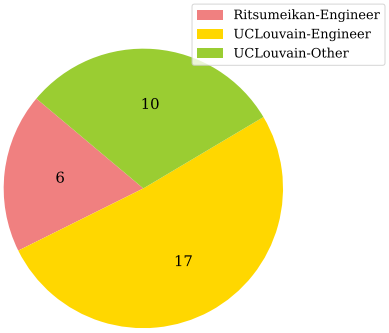
(i) Pick one of the furthest blocks from the robot

Figure 8.9: All the results for the change of objective method. In red, the decision of GPT-4 and in the colormap, the answers from the users obtained through a user study.

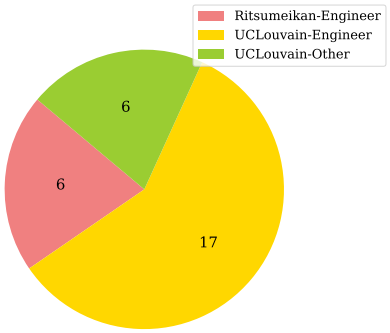
8.5 Experiment Participant Distribution



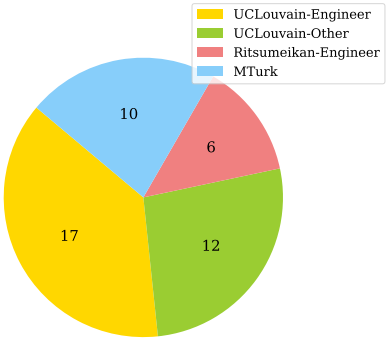
(a) GPT Factor: Experiment 4.2



(b) Safety Evaluation: Experiment 4.3



(c) Withdrawal: Experiment 4.5



(d) Changing Objective: Experiment 4.5

Figure 8.10: Number of participants and their distribution for each user study realized in this thesis.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**