School of Mechanical and Manufacturing Engineering
Faculty of Engineering
UNSW Sydney

BY

Nicholas James Bell

Real-Time ChatGPT Hybrid Decision-Making for Safe Human Robot Interaction

Thesis submitted as a requirement for the degree of Bachelor of Engineering (Honours) (Robotics and Mechatronics)

Submitted: 15 November 2024	Student zID: z5364122
Supervisor: Jose Guivant (UNSW)	

ORIGINALITY STATEMENT

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed: Nicholas Bell
Date: 15 November 2024

Abstract

This project aims to reduce communication delays caused by integrating Large Language Models (LLMs) into robot decision systems by instead combining deliberative and reactive strategies to form a hybrid decision-making model. This model benefits improved trust in human-robot interactions, operational efficiency, and real-time adaptive control. Despite successful research in the individual fields of hybrid models and LLMs, there remains limited exploration in effectively merging these components into one cohesive model. The report outlines literature research and a future project plan that involves refining the hybrid model and validating it through simulations and real-world robotic tests.

Acknowledgments

I want to extend a big thank you to my supervisor, Dr. Jose Guivant, for all the support and guidance he has given me this last year leading up to writing this thesis. It has been a great journey of learning and a challenge to keep up in this fast-growing research subject. However, Dr. Guivant has helped keep me grounded to realistic goals and always sought to help me succeed.

Contents

Abstract **3**

Acknowledgments **4**

List of Figures **7**

Nomenclature **8**

1 Introduction **9**

2 Literature Review **10**

2.1 Background History 10
2.2 Hybrid Framework 10
2.2.1 Overview of Hybrid Decision-making Systems 10
2.2.2 Enhancing Hybrid Systems with LLMs 11
2.3 Human-Robot Trust 11
2.3.1 Trust in Robots 11
2.3.2 Human Operator on the Loop 12
2.4 LLMs in Robotic Systems 12
2.4.1 ChatGPT APIs 12
2.4.2 Prompts 14
2.4.3 Zero-shot vs Few-shot Learning 15
2.5 LLM Decision Methods 15
2.5.1 Curriculum Learning 15
2.5.2 Perception-Action Feedback Loop 16
2.6 Gaps in Literature 16

3 Preliminary Research **17**

3.1 MATLAB 2D Simulation and Bug2 Algorithm 17
3.2 ChatGPT API MATLAB Integration 18
3.3 Transition from MATLAB to ROS2 & MoveIt 19

4 Methodology **20**

4.1 System Architecture Overview 20
4.1.1 Hardware Setup 20
4.1.2 ROS2 Software Framework 21
4.1.3 Workspace Structure 21
4.1.4 Hybrid Planning Architecture 22
4.2 Implementation Stages 23
4.2.1 Stage 1: Setup and Calibration of the UR5e Robot and ROS 23
4.2.2 Stage 2: Setup 3D Depth Camera and Object Detection 23
4.2.3 Stage 3: Reactive Path Planning with MoveIt 24

4.2.4 Stage 4: Developing ChatGPT as the Deliberative Task Planner 24

5 Results and Discussion **26**

5.1 Simulation Results 26
5.1.1 Human Detection and Interaction 26
5.1.2 3D Simulation in MoveIt 27
5.2 Real-World Testing Results 28
5.2.1 Global Planner Performance 28
5.2.2 Hybrid Planner Expectations 28
5.3 Comparative Analysis 29
5.4 Democratizing Robotics: Opportunities and Challenges 29
5.5 Reactive Safety Measures: Addressing the Risks of Democratized Robotics 29
5.6 Expanded Impact of Work in Real-World Applications 30
5.6.1 Industrial Applications 30
5.7 Healthcare Applications 30
5.8 Collaborative Robotics in Customer Service 30

6 Conclusion **31**

6.1 Contributions to Research Gaps 31
6.2 Implications for Human-Robot Collaboration 31
6.3 Limitations 31
6.4 Future Work 32
6.5 Final Remarks 32

References **33**

Appendix **35**

List of Figures

1 Hybrid Planning Architecture focused . . . 11
2 ChatGPT 4o-mini API pricing 2024 . . . 13
3 Basic Prompt describing the robots environment, state, constraints and requirements . . 15
4 Bug2 Obstacle avoidance 1 . . . 17
5 Bug2 Obstacle avoidance 2 . . . 17
6 Basic taskplan . . . 18
7 JSON file structure . . . 18
8 Randomly generated object map . . . 18
9 Executed MATLAB task plan . . . 18
10 Custom ChatGPT prompt for MATLAB experiment . . . 19
11 Hybrid Planning Architecture Overview . . . 22
12 UR5e Robot arm visualized in Rviz2 . . . 23
13 Custom prompt for UR5e Hybrid Planning . . . 25
14 Yolov8 Human Pose Detection . . . 26
15 UR5e obstacle collision avoidance inside MoveIt . . . 27
16 Real-world UR5e Global planner . . . 28
17 Gantt Chart Plan . . . 35

Nomenclature

AI Artificial Intelligence

API Application Programming Interface

LLM Large Language Model

NLP Natural Language Processing

$UR5e$ Universal Robot's e-Series Robotic Arm

$URDF$ Unified Robot Description Format

1 Introduction

The past two years have witnessed tremendous growth in research related to artificial intelligence (AI), with significant development towards large language models (LLM), such as OpenAI's ChatGPT, Google's Palm-E, and Meta's Llama 3. These developments mark the beginning of a paradigm shift in programming and robotics, expanding the capabilities and potential of machines to understand and generate human-like language in a dialogue format [4].

As robots and AI technology continue to advance and become more integrated in society, the need for trust between human-robot interactions becomes paramount and a necessary step in the right direction, especially in future environments where robots work in close proximity to humans. [19]. Large language models, due to their robust reasoning abilities, present a promising opportunity for enhancing the contextual awareness and decision-making processes of robots. However, a key challenge awaits: designing frameworks that integrate LLMs into robotic systems while prioritising safety. Fortunately, with the rampant competition and every newly released LLM, the models become faster and more efficient and, in turn, more reliable.

This report aims to explore the integration of LLMs, particularly ChatGPT, within robotic control frameworks that incorporate a hybrid behaviour selection system of both reactive and deliberative decision-making strategies [9]. This goal-based decision-making method was designed to simplify the decision model process and target the knowledge gap in reducing communication delays in LLM robotic systems. The hybrid decision-making simultaneously ensures that the robot retains the capability to swiftly respond to immediate sensory inputs through preprogrammed reflex actions, whilst also utilising informed planning facilitated by ChatGPT as an LLM. This hybrid method enables a robotic system to adapt and use reactive fallback strategies when faced with uncertain or unanticipated scenarios to ensure continuous operational flow. Furthermore, the hybrid method prioritises human safety by incorporating mechanisms for collision avoidance, dynamic replanning, and continuous feedback processing to ensure seamless and secure interactions with human operators. This is crucial to maintaining the efficacy and safety of robotic applications, especially in environments where the robot is required to collaborate closely with humans.

The primary objective of this work is to advance robotic systems that can safely and effectively collaborate with humans, enhancing trust through increased adaptability and reliability. The report begins by exploring the current state of the field through a literature review, identifying knowledge gaps. It outlines the design, implementation, and validation of a robotic control system that utilises hybrid decision-making with the integration of ChatGPT task planning.

2 Literature Review

This section reviews key research areas related to the integration of large language models (LLMs) in robotics, addressing the research field's rapid evolution, human-robot trust, decision-making frameworks, and existing research gaps.

2.1 Background History

Natural language processing (NPL) advancements made significant progress in human-robot communication, enabling machines to interpret natural language, ultimately leading to the development of LLMs. Similar to the space race, the AI race began in a hurry, and soon every major tech company had embarked on developing their own LLMs in contest for top performance and market dominance. However, not long after these developments did researchers realise that these models had remarkable capabilities for code generation from text prompts and pushed the boundaries of what these models were designed for by integrating them into robotic systems [17, 20, 3, 7, 11]. These studies, amongst many, have shown varying systems that have demonstrated the capability of translating ordinary command tasks into working machine code, with the aim of giving full control of a robot to non-technical users [17]. A notable study involved implementing ChatGPT into a robust hospital surgery robot called 'da Vinci', with the goal in mind of allowing ordinary doctors without prior operational training for the machine to easily use it via voice commands [12]. This advancement has begun to bridge the gap between human and robot interactions, creating a deeper level of trust and understanding by allowing a more human dialogue to be formed [19].

2.2 Hybrid Framework

2.2.1 Overview of Hybrid Decision-making Systems

Hybrid decision-making frameworks combine the immediacy of reactive control systems with a deliberative strategy and have been widely explored in robotics to enhance adaptability, with the capability of complex goal-oriented planning based on the perceived environment. Typically, as seen in Figure 1, reactive components handle instant sensor-based responses crucial for immediate actions, while deliberative components focus on processing gathered data to make informed decisions [13]. This combination allows the robot to react with quick reflexes, as well as having adaptable behaviours suitable for dynamic and uncertain environments [9].

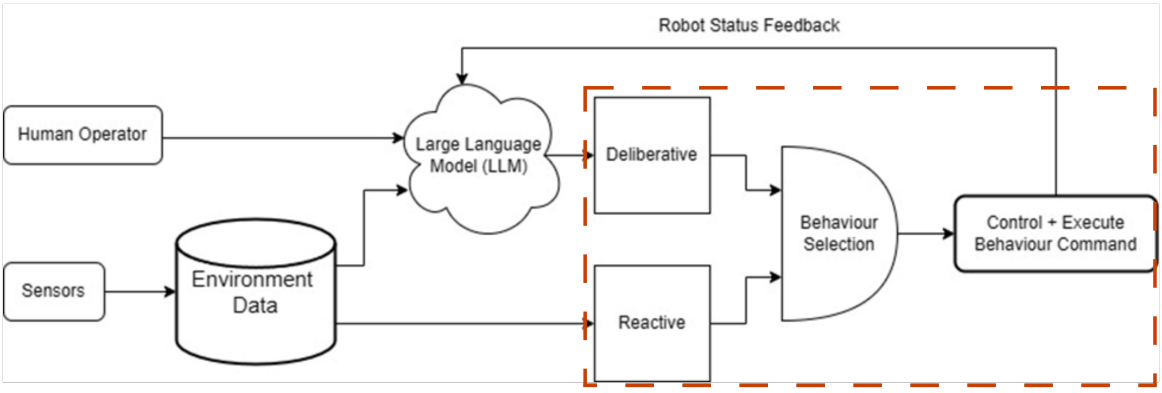


Figure 1: Hybrid Planning Architecture focused

2.2.2 Enhancing Hybrid Systems with LLMs

Integrating LLMs within hybrid systems enhances the decision-making capabilities and adaptive ability to respond to changes; however, it presents unique challenges due to increased computational loads that need to be managed to maintain the system's real-time responsiveness. As such, the hybrid system can perfectly balance this dynamic, such that when the LLM's processing times are delayed, the reactive systems can provide immediate actions. As seen in Figure 1, a behavioural selection component balances this dynamic, choosing between reactive and deliberative methods based on selection criteria and environmental conditions. This criterion is required to be further optimised but may include judging by processing times, user and robot safety when compared to immediate environmental changes, and sensor-based decisions.

The potential of hybrid systems resides in their capacity to enhance operational efficiency and safety in robotic applications, integrating the swift responses of reactive systems with the thorough decision-making of deliberative components. However, such integration presents significant challenges, including resolving decision conflicts and ensuring seamless communication between systems [17]. References to "Logic-Based Hybrid Decision System for a Multi-Robot Team" illustrate the necessity for balance, as these deliberative systems, while robust, require careful integration to avoid impairing real-time performance [12].

2.3 Human-Robot Trust

2.3.1 Trust in Robots

Reliability of robotic systems is fundamental when it comes to human trust, as people perceive robots performing in predictable and understandable ways. Communication is also a key factor for effective human-robot interactions, as it humanises an otherwise artificial machine, while delayed communication breaks down the illusion. However, only when human-like dialogue and robot reliability are merged do humans feel more confident in trusting robots [19]. This trust dynamic is accelerated with the integration of LLMs into robotics, enhancing human-robot dialogue, since trust is maintained and

strengthened by the accuracy and contextual appropriateness of robot responses [19].

Current research on trust in human-robot interaction frequently examines the complex nature of trust from a psychological approach [19] and generally overlooks the evaluation of trust from a robot reaction time perspective. However, it can be recognised that in an industrial environment, a faster work pace of a robot with a human controller increased mental stress [16]. This enlightens previous concepts as to how there is a limit to how fast this relation can operate, as currently the strain rests on LLM systems but may eventually shift to human operators.

2.3.2 Human Operator on the Loop

Integrating a human operator into the loop of robotic control systems, as seen in Figure 1, leverages human experience to supervise and guide robot actions and improves robot learning through direct human feedback in LLM models [19], [14]. However, the role of the human operator is evolving with advances in LLM technologies. As robots become more autonomous and capable of understanding complex instructions, the human operator's role shifts from direct control to more of a supervisory and corrective capacity. This approach is vital in contexts where full automation is impractical or unsafe, ensuring that LLM robots operate in a way that meets human standards without compromising safety or efficiency [14]. This dynamic is visualised in Figure 1, as the human operator is capable of overseeing the LLM as well as acting inside the feedback loop, providing new information to the model. This promotes human-robot trust, as it increases the reliability of robot action now that an overseer can ensure that any faults are well contained.

2.4 LLMs in Robotic Systems

2.4.1 ChatGPT APIs

With the rapid development of LLMs, it becomes necessary to use Application Programming Interface (API) systems as a method to seamlessly integrate LLMs into robotic systems. These systems facilitate the seamless integration of LLMs into robotic frameworks, allowing robots to interpret human directives, analyse contextual environmental data, and generate actionable task plans. These API commands call to high-level preprogrammed code blocks within the robot, such as 'move(object)', 'look_for(object)' and 'pick_up(object)'. The robot is able to convert high-level natural language commands into low-level robot control commands, allowing the LLM to use these APIs as a toolbox of commands, which may be further combined to form higher-level task combinations [17].

The use of APIs also allows for the potential of switching between various LLM models, as the robot's high-level functions are not designed specifically for any one system. This proves beneficial when the release of improved models necessitates a change to achieve better outcomes.

One of the main limitations when integrating ChatGPT via its API is the token limit per request. A token can be a word or a portion of a word, and ChatGPT's API allows a specific number of tokens per input and output based on the model version. The current chatgpt4-turbo allows a maximum

input context window of up to 128,000 tokens and a 4096 token output limit. These limits were designed to allow for an extensive input dataset while yielding a precise and focused output. Due to this, command structuring must be precise but concise to maximise every interaction within these limits. This token system is important since it impacts the model's capacity to comprehend long instructions or to extend without requiring further API calls. The integration of these models in robotics is limited to this constraint, but by creating a dialogue with the operator and splitting the input into smaller sections, it is possible to avoid reaching this limit [17].

The cost associated with utilising the ChatGPT API represents a significant consideration. As seen in Figure 2, using cloud-based LLMs like ChatGPT-4o can incur significant fees for API usage based on the quantity of tokens processed, indicating that frequent or complex requests may result in large token usage and high operational costs.

Alternatively, running a local open-sourced LLM such as Llama 3 via Ollama demonstrates promising results using an API without incurring server fees. However, it was found to incur high electricity costs to run the server locally or even with online cloud services, rendering it impractical for long-term development. Recently, ChatGPT released a new model called ChatGPT-4o-mini, which offers significantly cheaper and more affordable API pricing, nearly a quarter of the cost of the standard ChatGPT-4o model, as seen in Figure 2. This development presents a viable real-time communication solution for the future phases of the project, balancing both performance and cost-effectiveness.

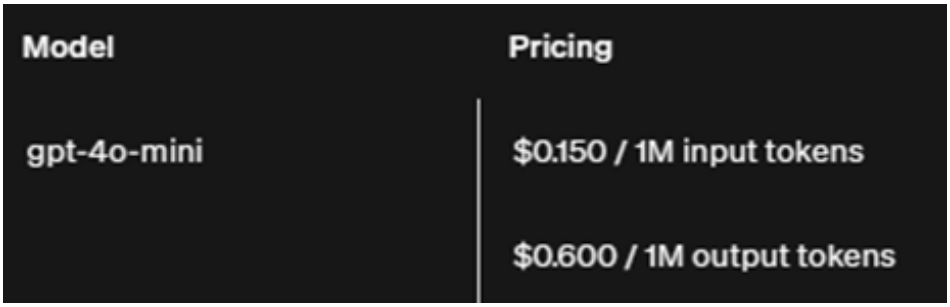


Figure 2: ChatGPT 4o-mini API pricing 2024

For budget-conscious projects such as this, it requires careful planning and optimisation of interactions with the API to balance functionality with financial feasibility. To effectively manage costs while maintaining rapid performance, selected strategies may be used:

- **Optimising Language Use:** Streamline commands to reduce token usage without losing the essential details necessary for task execution.
- **Selective Interaction:** Limit real-time interactions to essential queries and commands, using preprocessed data or cached responses when possible.

Alternatively, it is possible to directly input the prompt and command into standard ChatGPT, followed by manually entering the results into the robot system, bypassing the need for an API system, and it may be more beneficial during the early model development. As this AI race continues

and models evolve, the cost and efficiency of using LLMs like ChatGPT in robotics are likely to improve, making it more accessible for broader applications, including academic research and commercial robotics.

2.4.2 Prompts

LLMs in robotics go beyond the standard use cases of language translation or text generation by utilising them for physical tasks that require adaptation of real-world knowledge [1]. Terms like prompts, API calls, and zero-shot learning are commonly used in related literature to describe the many approaches used to command and control robotic behaviours using language models. Prompt engineering, especially, has emerged as an art form where the precise wording of instructions can significantly affect the effectiveness of the robot's actions.

Formulating prompts for LLMs is a sophisticated methodology for instructing robots on how to engage with their surroundings and execute tasks. A prompt's structure plays a crucial role in dictating the LLM's decision-making process. This is agreed upon in related research as crucial for guiding the model to yield a desired response when each word is acting as a directive to shape the model's understanding. This precision in prompt design is increasingly essential as researchers strive to communicate complex, multi-step tasks to robotic systems in a language that they can operate effectively.

The challenge of integrating LLMs into robotics is the issue of input prompt ambiguity, followed by the model's tendency to diverge greatly with interpretations and hypothetical results. To address this, there has been significant progress in developing standardised prompt libraries and establishing recommended methods for prompt creation [5]. These efforts aim to ensure a higher degree of predictability and consistency in robot behaviour, which is essential for human-robot trust. Related literature provides foundational practices in prompt engineering, but there is a noticeable gap on how various prompt designs affect the accuracy of how the commands are interpreted.

Different prompt designs are very important for outlining how the LLM will respond to commands and how they may use the information provided to accomplish assigned tasks. A basic curriculum learning approach is outlined in Figure 3, which utilised high-level API functions to act as a toolbox for the learning model, allowing it to appropriately create a task plan to accomplish its goal [17]. These functions must be named, explicitly outlining the task of the function, allowing the learning model to accomplish higher success rates whilst interpreting user commands [19].

```
Imagine you are programming a robot arm whose main task is to move a block. The space contains several items: a red block,
a blue block. The functions available for controlling the robot include:

get_coordinates(object_name): Returns the XYZ coordinates of the object.
move_to(object_name): Moves the robot to a location determined by the XYZ coordinates of the given object and does not
return any value.
pick_up(object_name): Instructs the robot to pick up the object and does not return any value.
put_down(object_name): Instructs the robot to put down the object and does not return any value.

You are not to use any other hypothetical functions.
Using these functions, can you script a sequence that would enable the robot to pick up the red block and put down next to
the blue block?
```

Figure 3: Basic Prompt describing the robots environment, state, constraints and requirements

2.4.3 Zero-shot vs Few-shot Learning

The robotics field has embraced the concepts of zero-shot and few-shot learning, in the pursuit of greater robot adaptability. These concepts enable LLMs to perform tasks without prior or minimal exposure. Specifically, zero-shot learning shows that it can generalise from its pre-trained knowledge to form new tasks [7], while few-shot learning allows robots to quickly adapt to new tasks with minimal examples, learning so that robots may quickly adjust to new settings or instructions [18]. Zero-shot and few-shot learning are valuable for tasks requiring rapid adaptation to new situations with minimal prior data.

2.5 LLM Decision Methods

The primary goal of integrating large language models (LLMs) such as ChatGPT in robotics is to bring a human operator into the loop without requiring the human operator to have coding experience. Accomplishing this would expand the human and robot-related research into LLM models' approaches on how to use the operator differently, but the core concept of having them within the loop as supervisors and commanders remains. Basic model ideas began with an open-loop command curriculum learning approach [3], moving on to perception-action feedback loops and finally to real-time adaptive models [17].

2.5.1 Curriculum Learning

Curriculum learning for an LLM in robotics is inspired by the structured approach used in education systems. It involves teaching the robot small-scale skills represented as APIs, which can be further combined to challenge more complex tasks [17] and is used in the prompt in Figure 3. This approach makes use of ChatGPT's adaptability to introduce new ideas or abilities, enabling the robot to gradually expand on its base knowledge in a controlled manner.

The related literature highlights the effectiveness of this approach in reducing training time for complex tasks, especially when combined with the capabilities of LLM to contextualise and extrapolate from limited data. However, this has the possibility of leading to hallucinations and for the model to create false hypothetical API skills, where there are none within the system, and as such, it is necessary for the robot to first check if the model commands exist and are used within limits [17], [7]. As seen in

Figure 3, the prompt explicitly says 'You are not to use any other hypothetical functions' in reference to the model, and although this is somewhat successful, it does not fully rule out all instances where the model has hallucinated.

2.5.2 Perception-Action Feedback Loop

LLMs lack direct perception of their environment and require external sensor information to be parsed in the form of semantic text descriptions to better base decisions [3], [7]. This is particularly challenging in dynamic environments where external factors are unpredictable. These limitations can be overcome at a foundational level in robot architecture by using alternative decision-making methods. Methods, such as reinforcement learning or behaviour trees, enable LLMs to learn from past interactions and improve their adaptive control and ability to navigate environments or detect external disturbances [14], [21].

Alternatively, ChatGPT can form a closed feedback loop where perception information is input as text in a dialogue format and used to inform decision-making. Perception-action feedback loops greatly enhance robotic systems, allowing for adaptation based on environmental interactions, which has proven difficult for basic curriculum learning. These loops are crucial for robots to perceive their surroundings, make informed decisions, and act on those decisions.

2.6 Gaps in Literature

The research shows ongoing gaps in the integration of large language models (LLMs) such as ChatGPT into real-time robotic systems, despite significant progress. Although the theoretical benefits of hybrid decision-making frameworks are well acknowledged, their integration with a large language model is relatively uncommon. A research gap stands to thoroughly assess the integration of a robot's reactive capabilities with the strategic planning provided by LLMs such as ChatGPT. The focus will be on the performance of these integrated systems under varied operational conditions, examining how they balance sensor-driven reactions with AI-informed decision-making [9].

Moreover, this study will offer a thorough analysis of the ways in which ChatGPT's deliberative thinking and reactionary reflexes combine to develop hybrid models, highlighting both their advantages and disadvantages for practical robotic applications. The objective is to establish a library of best practices that will enable the smooth and operationally efficient integration of LLMs such as ChatGPT with reactive control mechanisms. This analysis will help strengthen techniques for hybrid decision-making.

Furthermore, although the development of hybrid decision-making systems is notable, thorough research on the interaction between various components of the system and their combined effect on system performance is still lacking in the literature [9]. This shortfall implies that a more thorough study of the dynamics of these systems is required.

These gaps are sometimes filled with excessively generalised and non-practical specificity [9]. This research will create doable plans meant to close these gaps. It will concentrate on using these techniques within the framework of employing ChatGPT, making sure that the answers are workable theoretically and practically.

This research will assess and optimise integrating LLMs into robotic systems within a hybrid framework. This aims to guarantee that robots equipped with LLMs can attain both the prompt responses required for reflexive actions and the intricate decision-making, thereby enhancing the efficiency and reliability of robotic operations in dynamic settings.

3 Preliminary Research

The preliminary stages of this research were essential in developing a fundamental understanding of hybrid decision-making systems and the methods of integrating ChatGPT APIs with custom prompts. The insights gained during this stage directly informed the migration away from MATLAB and into a more advanced real-time ROS and MoveIt environment to address the project's specific goals.

3.1 MATLAB 2D Simulation and Bug2 Algorithm

These preliminary stages replicated the success of Microsoft's and SayCan's implementation of an LLM with MATLAB to experiment with this new robot control method [17]. A 2D mobile robot simulation was developed inside MATLAB to test the reactive component of the decision-making system, implemented using the Bug2 obstacle avoidance algorithm. This enabled the simulated robot to avoid obstacles by tracing their boundaries and recalculating its path to the target. This algorithm is designed to follow the border clockwise around the obstacle starting on its left, tracing the robot's completed path in green as seen in Figure 4 and Figure 5.

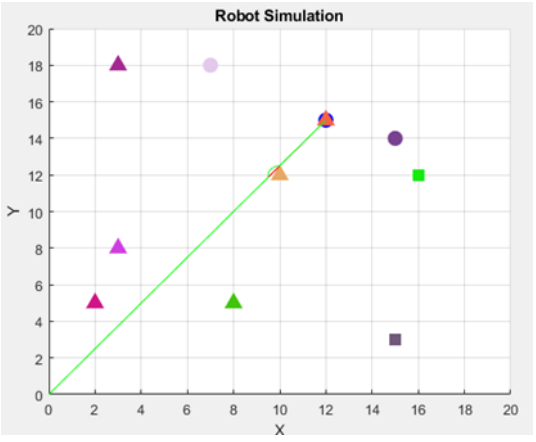


Figure 4: Bug2 Obstacle avoidance 1

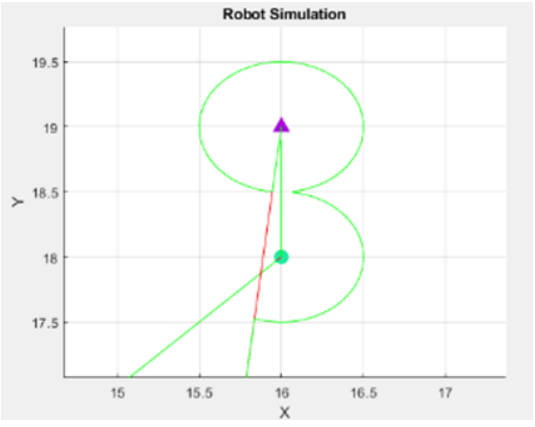


Figure 5: Bug2 Obstacle avoidance 2

Although the algorithm is inherently simple, its predictability provided a reliable foundation for understanding the nuances of reactive planning, particularly in contexts where safety is paramount. This stage validated the importance of a reactive mechanism that could respond to environmental changes, a principle later adapted to MoveIt's hybrid planning framework [9, 15].

3.2 ChatGPT API MATLAB Integration

The deliberative component of the architecture was tested by integrating ChatGPT to interpret and parse high-level natural language commands in Figure 6 and generate actionable steps for the robot in a JSON format as per Figure 7.

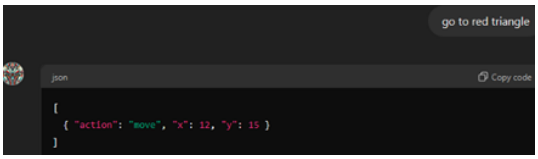


Figure 6: Basic taskplan

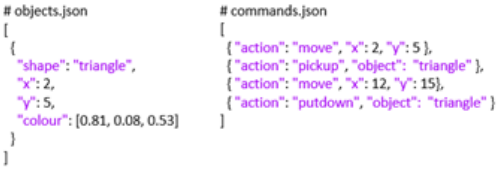


Figure 7: JSON file structure

Utilising prompt engineering, ChatGPT converted user command requests into a series of JSON commands, which were executed in the MATLAB environment. For instance, commands such as "go to red triangle" in Figure 6, were successfully parsed into precise task plans in Figure 7. Challenges surrounding ambiguities in object identification were mitigated by instructing ChatGPT to incorporate secondary identifiers like colour and shape as described in the semantic description file, ensuring task accuracy [17]. For this test, object identification is simulated by generating a *objects.json* file during the map creation process of the 2D robot simulator.

Further testing showed the command parsing and strategic task planning capabilities of ChatGPT generated and executed complex task plans based on the updated state information. A custom prompt was given the object location on a random map in Figure 8 and then instructed to "Go to every object in the most efficient path." ChatGPT responded by generating a command JSON file, successfully instructing the robot to move to every object as demonstrated by the green line in Figure 9.

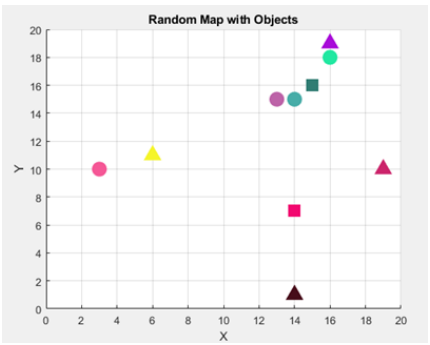


Figure 8: Randomly generated object map

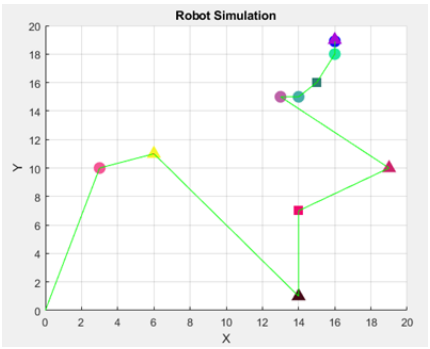


Figure 9: Executed MATLAB task plan

Experimenting with various prompts found that by providing an example it performed better according to experiments. The prompt used to accomplish this task is documented below:

```
You are controlling a robot that can perform actions such as moving to specific coordinates, picking up objects, and dropping objects. The robot's actions need to be described in a JSON format. Each command should include the action to be performed and the necessary parameters.

Here are the available actions and their parameters:
1. "move": The robot moves to a specific coordinate.
- Parameters: "x" (x-coordinate), "y" (y-coordinate)

2. "pickup": The robot picks up an object. It stays picked up until dropped.
- Parameters: "object" (name of the object to pick up)

3. "drop": The robot drops an object.
- Parameters: "object" (name of the object to drop)

Here is additional information:
1. The grid size has a size of [0 20 0 20]
2. The Initial robot position or home is, [0, 0]

A JSON file named objects.json is uploaded and it must be used as a semantic vision guide. The format of the JSON file is as follows:
[
{"shape": "triangle", "x": 16, "y": 20, "colour": [0.88, 0.12, 0.36]},
{"shape": "triangle", "x": 9, "y": 3, "colour": [0.68, 0.41, 0.62]}
]

In the case where object identifiers are repeated in the objects.json file and the command does not specify which object, you may ask for further clarification based on the available identifiers; if colour is available, use this as the alternative identifier, and convert the RGB value to the actual colour in english.

If a new objects.json is uploaded, ignore the old json file and only use the new version.

Please provide a sequence of commands for the robot in the following JSON format:
[
{ "action": "move", "x": 10, "y": 5 },
{ "action": "pickup", "object": "block" },
{ "action": "move", "x": 15, "y": 10 },
{ "action": "drop", "object": "block" }
]

You may only write ONE single sentence of any additional text, clarification or explanation. Then provide the JSON output code file.
```

Figure 10: Custom ChatGPT prompt for MATLAB experiment

3.3 Transition from MATLAB to ROS2 & MoveIt

The progress made for the preliminary research demonstrated the feasibility and effectiveness of integrating ChatGPT with a robot system for hybrid decision-making. The successful implementation of the Bug2 obstacle avoidance algorithm and the accurate execution of commands highlight the potential for this system to handle complex tasks in dynamic environments. The 2D simulation provides a solid foundation for future developments, including the integration of the Primesense Carmine 3D camera for enhanced perception and interaction capabilities.

While these preliminary efforts confirmed the feasibility of combining deliberative and reactive components, they also revealed significant limitations of the MATLAB-based approach. The 2D environment was incapable of simulating real-time dynamic interactions or integrating intricate sensor data, including 3D human poses. The lack of advanced motion planning frameworks underscored the necessity for a move to a more robust system adept at managing real-world situations.

The research carried out using MATLAB established an essential understanding of reactive and deliberative decision-making, which directly impacted the design and execution of the ROS2-based system. The Bug2 algorithm's demonstration of dependable reactive behaviour influenced the setup of reactive planning in MoveIt, while the ChatGPT-based task planning confirmed the effectiveness of natural language processing in producing actionable commands. Transitioning to ROS2 enabled the system to function in a 3-D environment, manage real-time inputs, and incorporate advanced planning and detection frameworks.

4 Methodology

The main objective of this project is to develop a hybrid decision-making architecture that explores real-time integration of LLMs, specifically ChatGPT, into dynamic robot environments that allow a robot to execute complex tasks issued by human directives while prioritising safe human-robot interactions. This methodology outlines the approach and development process that resulted in a functional simulation of this goal, achieved through the implementation and integration of key tools and techniques, as informed by existing literature [4, 17, 9].

4.1 System Architecture Overview

The robotic system is built utilising a ROS2 framework, which integrates multiple components to enable efficient communication. The system utilises an UR5e robotic arm controlled via ChatGPT commands as an advanced task planner, an adapted Moveit hybrid path planner for movement, and a 3D depth camera for human detection. Each component plays a vital role to facilitate safe and flexible human-robot collaboration [12, 17].

4.1.1 Hardware Setup

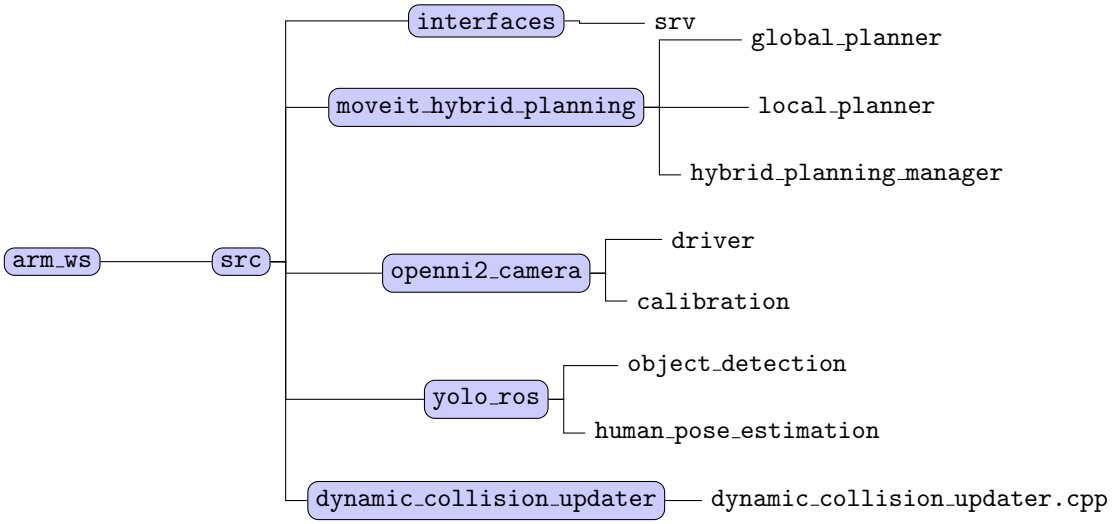
- **UR5e Robotic Arm**: The UR5e is a 6-DOF robotic arm that provides full manoeuvrability within its workspace. This robotic arm was supplied by the lab and serves as the primary manipulator for task execution.
- **PrimeSense RD1.09 3D Depth Camera**: An older yet reliable depth camera, the PrimeSense RD1.09, was available and used for human detection. The $openni2_camera$ wrapper was necessary to port the older generation camera into ROS2 framework, and despite its challenges in implementing it, it was successfully integrated with the system. [14]. The $openni2_camera$ wrapper is owned by a third party and publicly available on GitHub [2].

4.1.2 ROS2 Software Framework

The ROS2 framework manages communication across multiple packages and nodes, enabling seamless integration of various subsystems.

- **YOLO_ROS Wrapper**: A specialised adaptation of the YOLO object detection application for ROS2, this wrapper is responsible for detecting humans and objects in real time. The positional data generated is used to create collision objects within the MoveIt planning scene. This wrapper is owned by a 3rd party and publicly available on GitHub [6].
- **ChatGPT**: ChatGPT acts as the high-level task planner, interpreting natural language commands and translating them into executable task plans. This integration enables intuitive user interaction and flexible task execution [17, 8].
- **MoveIt Hybrid Planning**: The hybrid planner within MoveIt provides reactive planning capabilities, allowing real-time path adjustments to avoid collisions. This component ensures the robot operates safely in dynamic environments [9, 3]. The kinematic path planning solver used by default for the UR5e robot was $kdl_kinematics$. The package is owned by MoveIt and publicly available [10].
- **Dynamic Collision Marker Updater**: The marker updater dynamically creates and updates collision markers in the MoveIt planning scene. These markers are based on the YOLO human pose marker array, ensuring accurate real-time adjustments to the planning environment.

4.1.3 Workspace Structure



4.1.4 Hybrid Planning Architecture

This hybrid planning architecture, as shown in Figure 11, combines inputs from human operators and the 3D depth camera for object detection sensing, processed by both the LLM and reactive modules. The behaviour selection unit is controlled inside the MoveIt hybrid planning package, an existing framework developed by MoveIt, and accepts movement requests from the task planner [10].

This approach integrates a control manager that utilises two asynchronous path planners, operating independently of each other, to balance long-term planning with real-time adaptability. The control manager uses a global path planner that focuses on finding an overall path to the target, using the Open Motion Planning Library (OMPL) to plan around obstacles inside its workspace before the robot plan is executed. Similarly, a local path planner is employed to handle immediate obstacle avoidance or adapt to unforeseen changes on the trajectory path. According to the MoveIt documentation [10], hybrid planning is especially appropriate for situations necessitating secure human-robot interaction, wherein the robot must quickly adapt to unforeseen obstacles and environmental changes.

The robot status feedback, including information about the robot's current position and task progress, is then parsed back into ChatGPT with new environment data, awaiting human operator command. This feedback loop is essential for maintaining an up-to-date understanding of the robot's state within its environment, as seen in Figure 11.

This interactive loop enables the human operator to issue high-level commands informed by the robot's current state and environmental context. For instance, if the feedback indicates that the reactive planner is stuck or unsolvable, the operator can direct the robot to reroute or take corrective measures. Alternatively, if the robot has successfully completed a task, the operator may issue a new command for the next task, ensuring seamless continuity in a closed-loop system.

This structure promotes a human-centred approach to robotic control, where humans and robots work together in real time, enhancing both the safety and efficiency of human-robot interaction.

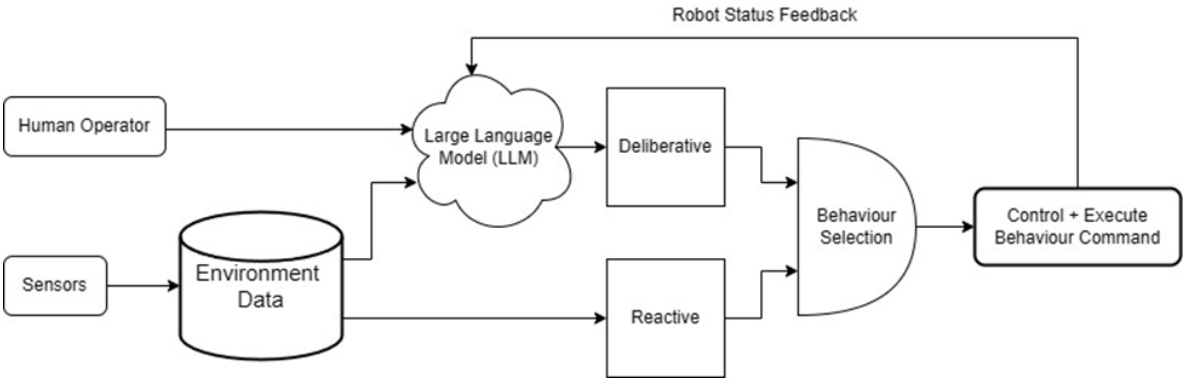


Figure 11: Hybrid Planning Architecture Overview

4.2 Implementation Stages

The implementation of the hybrid decision-making system was divided into structured stages to facilitate clarity and ensure systematic development.

4.2.1 Stage 1: Setup and Calibration of the UR5e Robot and ROS

The first stage focused on configuring the UR5e robotic arm within the ROS2 environment. The robot's Unified Robot Description Format (URDF) file was launched with Moveit and visualised inside of Rviz2, where additional boundary objects were also created to act as safety stops, as seen in Figure 12.

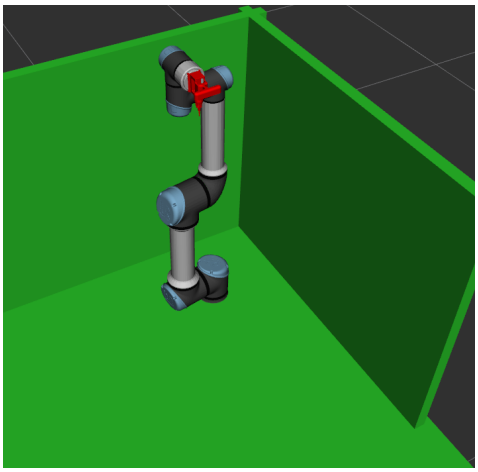


Figure 12: UR5e Robot arm visualized in Rviz2

As seen in red in Figure 12, there is an additional URDF file that was attached at a fixed point to the "tool0" flange of the robot arm, representing a custom end effector. This end effector was used for a similar project utilising the UR5e robot arm as an automated screwdriver system.

The integration of MoveIt provided the necessary framework for path planning and execution, as MoveIt actively takes into account the collision objects inside of this workspace in the path planning algorithm. The Rviz visualisation tool was used to validate the robot's motion and simulate planning algorithms using a fake UR5e controller. This effectively created a global path planner for the current path planning scene before locking the scene and executing motion.

4.2.2 Stage 2: Setup 3D Depth Camera and Object Detection

This stage emphasised integrating human detection capabilities provided by the YOLO ROS wrapper and the PrimeSense depth camera. The PrimeSense RD1.09 depth camera was mounted vertically above the workspace, and its transformation pose was calibrated from the base of the robot arm to detect objects and human poses accurately. The YOLO ROS wrapper processed this depth camera data in real-time, generating positional information that was converted into collision objects within the MoveIt planning environment using the custom dynamic collision marker updater package. These collision markers facilitated the robot's ability to respond dynamically to environmental changes, laying

the foundation for safe human-robot interactions [14, 17]. The 3D camera generated a point cloud, which was visualised inside of rviz.

4.2.3 Stage 3: Reactive Path Planning with MoveIt

To ensure safe and adaptive operation, the MoveIt hybrid planning framework was configured to combine global and local planners for real-time path adjustments. By dynamically updating collision markers inside the Moveit planning scene based on human poses, the system could respond adaptively to human presence, ensuring safety without compromising task efficiency.

4.2.4 Stage 4: Developing ChatGPT as the Deliberative Task Planner

The ChatGPT package simply executed a list of commands based off its task plan interpreted by the high-level natural language commands fed to it. This implementation leveraged both ROS2 service (srv) calls to act as a middleman between the robot control manager and the ChatGPT command interface. In this regard, the system can work entirely independently from the ChatGPT system, which is excellent for testing purposes but limits its capabilities and functionality focus in terms of this project. The ROS2 service interface facilitated seamless communication between the ChatGPT package and other components in the system, ensuring real-time interaction and providing a robust interface for human-robot collaboration.

Building upon the insights gained during the MATLAB phase, a refined prompt structure was designed to optimise the interaction, similar to what was learnt in the MATLAB experiment, to address ambiguities by incorporating object attributes such as colour, shape, and position. For example, commands like “move to the xyz” were translated into precise ROS2 service calls, ensuring accurate execution. Despite the hybrid planner utilising joint goal commands, the ChatGPT prompt was able to give specific joint goals based on the requested pose utilising inverse kinematics. However, a simpler, less computationally intensive approach would have been to import the inverse kinematic calculations inside the robot controller.

These service calls defined both the reactive and deliberative aspects of the hybrid decision-making system, enabling flexible and adaptive task planning. The transition to ROS2 enhanced responsiveness and reduced command latency, further aligning the system with real-time operational requirements [9, 11].

You are controlling a robot that can perform actions such as moving to specific joint positions, controlling its gripper, and adjusting its speed. The robot's actions need to be described in a JSON format. Each command should include the action to be performed and the necessary parameters.

Here are the available actions and their parameters:
1. "move_joints": The robot moves to specific joint positions.
  - Parameters: "joint_positions" (array of joint angles in radians, e.g., [0.0, -1.57, 1.0, 0.5, -0.5, 0.0])

2. "grip": The robot operates the gripper.
  - Parameters: "state" (open or close)

3. "adjust_speed": Adjusts the movement speed of the robot.
  - Parameters: "speed" (value between 0.1 and 1.0, where 1.0 is the fastest)

Here is additional information:
1. The robot's initial position, or home, is [0.0, -1.57, 0.0, 0.0, 0.0, 0.0].
2. Commands must include valid joint angles within the robot's operational range, or the system will reject the command.

A JSON file named joints.json is uploaded and must be used to determine valid joint angles for the robot. The format of the JSON file is as follows:

```
{
  "joints_range": [
    {"joint": "shoulder_pan", "min": -3.14, "max": 3.14},
    {"joint": "shoulder_lift", "min": -2.0, "max": 2.0},
    {"joint": "elbow", "min": -3.14, "max": 3.14},
    {"joint": "wrist_1", "min": -3.14, "max": 3.14},
    {"joint": "wrist_2", "min": -3.14, "max": 3.14},
    {"joint": "wrist_3", "min": -3.14, "max": 3.14}
  ]
}
```

In the case of conflicting or ambiguous commands, clarify based on the range values in joints.json. If a new joints.json is uploaded, ignore the old file and only use the new version.

Please provide a sequence of commands for the robot in the following JSON format:

```
[
  { "action": "move_joints", "joint_positions": [0.5, -1.0, 1.2, 0.8, -0.8, 0.3] },
  { "action": "grip", "state": "close" },
  { "action": "adjust_speed", "speed": 0.5 },
  { "action": "move_joints", "joint_positions": [1.0, -0.5, 1.5, 0.6, -1.0, 0.0] }
]
```

You may only write ONE single sentence of any additional text, clarification, or explanation. Then provide the JSON output code file.

Figure 13: Custom prompt for UR5e Hybrid Planning

5 Results and Discussion

This section documents the main findings found through extensive testing in simulated environments and real-world setups. Where real-world testing was incomplete, simulation results serve as reliable indicators, given their strong, accurate portrayal of the expected physical system behaviour. These results, supported by simulations and partial real-world testing, reflect the potential of democratising robotics through AI while emphasising the importance of implementing safety measures to address the inherent risks.

5.1 Simulation Results

5.1.1 Human Detection and Interaction

The integration of the Primesense depth camera and YOLO allowed the system to detect and avoid humans by generating collision objects around detected human joints.

- **Qualitative Feedback**: The system adapted effectively to new human positions, demonstrating robust real-time interaction in a collaborative environment. However, the human pose points lagged behind the movement in the real world, especially with fast movements, which is a cause for concern because this lag directly impacts the speed at which the system can react to the human movements. Additionally, at certain angles where the human image is distorted or even when another object passes in front, the object detection flickers in and out as it fails to detect the human.

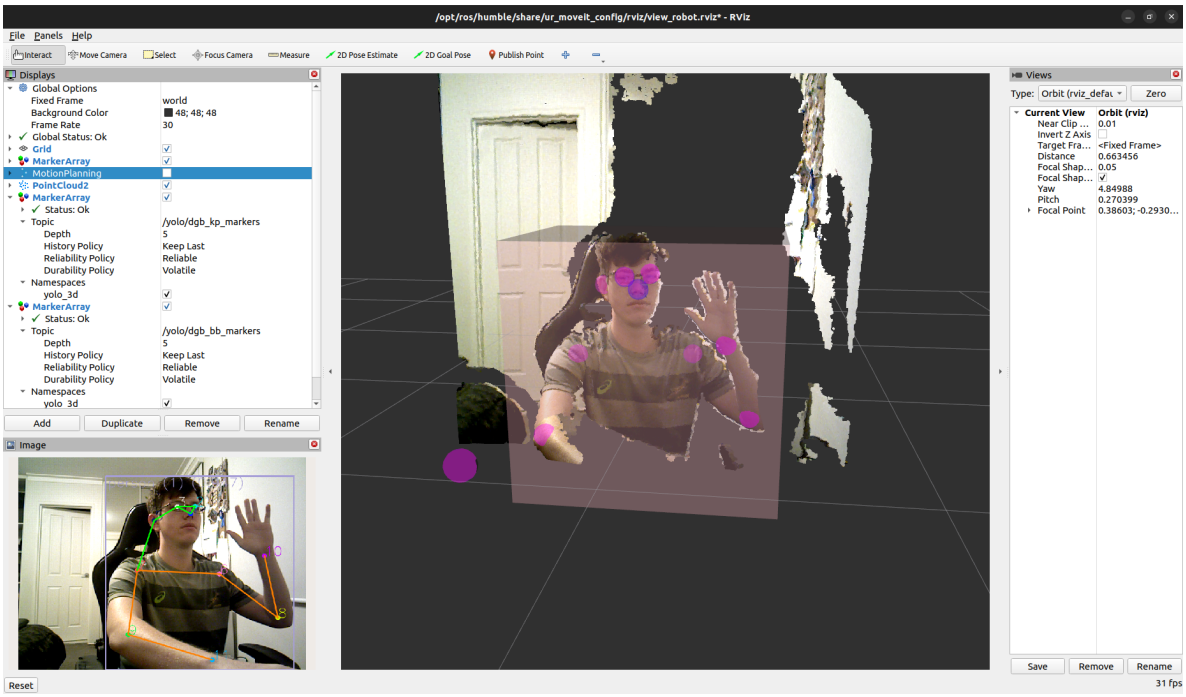


Figure 14: Yolov8 Human Pose Detection

5.1.2 3D Simulation in MoveIt

The hybrid planner was tested extensively in 3D environments using the ROS2 MoveIt framework. The MoveIt hybrid planning architecture enabled real-time path adjustments, allowing the robot to navigate dynamically changing environments. Simulation results demonstrated the hybrid planner's ability to detect and avoid moving obstacles, particularly those simulating human movement near the robot's workspace.

3D simulations in ROS and MoveIt allowed for more complex tests involving dynamically generated obstacles. Figure 15 illustrates the robot's path adjustments in response to obstacles.

- **Effectiveness of Hybrid Planning**: The MoveIt hybrid planner successfully adjusted the UR5e's path plan using its local planner in real-time when detecting objects or human presence along its global planner path plan.
- **Collision Avoidance with Human Proximity**: Safe distances were maintained in scenarios where a human operator approached the workspace, demonstrating the reactivity of the hybrid planning system.

However, in certain cases, the planner exhibited abrupt stopping behaviour when unable to resolve complex planning scenarios, such as when multiple moving obstacles created conflicting paths. In a sense, this is exactly how the robot should react when there is no solution found; it should simply stop to prevent collisions. The accuracy of these simulations makes them a reliable portrayal of the system's potential behaviour in real-world scenes.

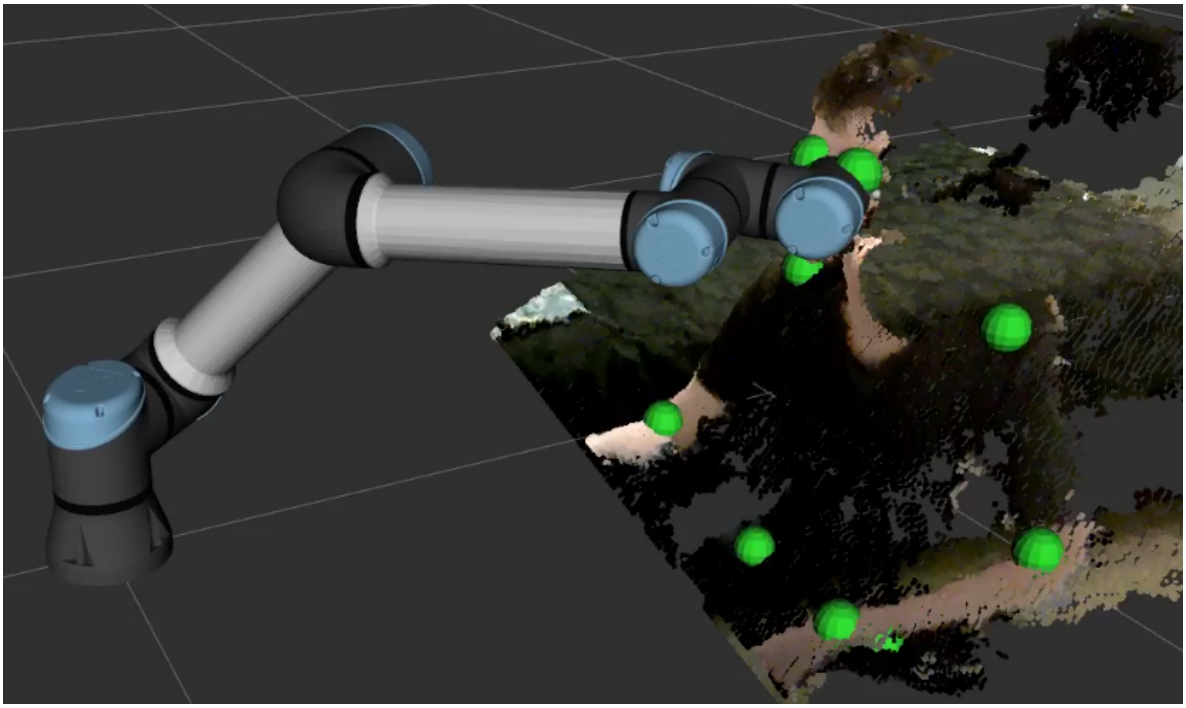


Figure 15: UR5e obstacle collision avoidance inside MoveIt

Figure 16: Real-world UR5e Global planner

5.2 Real-World Testing Results

5.2.1 Global Planner Performance

Real-world testing was conducted with the global planner, which demonstrated consistent performance in executing pre-defined tasks. The UR5e robotic arm successfully followed planned trajectories without deviations, accurately reaching target poses as specified in the commands, as seen in Figure 16.

The global planner's inability to replan in response to dynamic obstacles was a known limitation and was observed in scenarios where unexpected objects or human presence entered the robot's workspace. In such cases, the robot would pause or fail to complete the task safely. While this limitation underscores the necessity of the hybrid planner, the global planner proved effective for predictable, controlled environments.

5.2.2 Hybrid Planner Expectations

Due to technical constraints, the hybrid planner could not be tested in the real-world setup. However, based on the accuracy of the 3D simulations, it is reasonable to anticipate similar behaviour in physical environments. The simulations indicate that while the hybrid planner would perform well under typical scenarios, abrupt stopping behaviour in unresolved planning conflicts could pose a challenge in real-world dynamic settings.

5.3 Comparative Analysis

The results highlight the contrast between the global and hybrid planners. The global planner excels in pre-defined, static environments but lacks the adaptability required for dynamic interactions. In contrast, the hybrid planner addresses this adaptability but introduces complexities that can cause path planning failures.

The global planner demonstrated robust task execution in controlled environments. Commands were successfully interpreted and executed, allowing the UR5e robotic arm to complete pick-and-place tasks accurately. However, its limitations in handling dynamic obstacles highlighted the need for hybrid planning. When unexpected human presence or objects were introduced, the system lacked the adaptability required for safe operation, leading to task interruptions.

These observations reinforce the potential effectiveness of the hybrid planner in real-world setups. Simulation tests reliably indicated how the hybrid planner would perform under similar conditions, showcasing its superiority in maintaining safety and adaptability in dynamic workspaces.

5.4 Democratizing Robotics: Opportunities and Challenges

Recent advancements in AI, such as those highlighted by NVIDIA's and Microsoft's initiatives, position robotics as increasingly accessible to non-experts. Language-based interfaces like ChatGPT enable intuitive machine control, reducing the barriers for individuals without technical expertise. However, this democratisation introduces significant challenges, particularly for first-time users who may lack the situational awareness or experience necessary for safe and efficient operation.

In this project, ChatGPT acted as a high-level task planner, interpreting natural language commands and generating executable plans. This approach simplified robotic control and bridged the gap between user intent and robot action. However, simulations revealed potential risks of over-reliance on AI. Misinterpretations of commands or unforeseen environmental changes could lead to hazardous conditions. These findings highlight the critical need for reactive safety mechanisms to mitigate risks.

5.5 Reactive Safety Measures: Addressing the Risks of Democratized Robotics

The reactive safety measures implemented in this system represent a crucial step towards ensuring safe human-robot collaboration. By dynamically generating collision markers and adjusting paths in real-time, the hybrid planner effectively balanced task execution with safety considerations. These mechanisms acted as a safeguard, protecting both humans and the machine from potential accidents.

Figure 14 illustrates the system's human detection capabilities, which were central to this safety framework. The integration of the Primesense depth camera and YOLO ensured accurate real-time detection of human poses, enabling the planner to adapt its trajectory accordingly. These features are essential for preventing misuse or accidents in environments where first-time users interact with robots.

5.6 Expanded Impact of Work in Real-World Applications

The hybrid decision-making framework developed in this thesis has significant potential to impact various real-world applications, particularly in domains requiring safe, adaptive, and collaborative human-robot interactions. The integration of ChatGPT as a high-level task planner with reactive path planning through ROS MoveIt positions this system as a transformative approach to bridging the gap between intuitive human commands and real-time robotic adaptability.

5.6.1 Industrial Applications

In manufacturing environments, where efficiency and precision are paramount, this framework offers a means to enhance robotic versatility. Traditional industrial robots often follow pre-programmed paths that lack the ability to adapt to dynamic environments or human presence. The hybrid planner, with its ability to dynamically replan and respond to environmental changes, enables robots to operate safely alongside human workers. This capability can reduce downtime caused by environmental unpredictability, enhance productivity, and promote worker safety. For instance, in assembly lines where humans and robots must collaborate, the system can ensure that the robot adjusts its actions in real-time to avoid collisions while maintaining task efficiency.

5.7 Healthcare Applications

Healthcare settings often require robots to operate in close proximity to humans, such as in surgical assistance, elder care, or rehabilitation. The demonstrated ability to detect and adapt to human presence in real-time is critical for ensuring safety in such scenarios. For example, a robot assisting in a physical therapy session can adapt its motions dynamically based on the patient's movements, providing both safety and effective therapy. Similarly, robots equipped with this framework could assist in patient handling, medication delivery, or surgery, where precise and adaptive movements are essential to avoid harm.

5.8 Collaborative Robotics in Customer Service

In customer-facing roles, robots are increasingly being deployed to assist with tasks such as guiding customers in stores, delivering items, or answering queries. The integration of ChatGPT allows these robots to interpret and respond to natural language commands, making them more accessible and user-friendly. By incorporating reactive planning, the robots can navigate crowded environments, avoid obstacles, and engage with customers safely, enhancing their utility in service industries such as retail and hospitality.

6 Conclusion

This thesis has presented a hybrid decision-making framework that integrates ChatGPT for high-level task planning with reactive path planning in ROS MoveIt to enable safe and adaptable human-robot interaction. Using a UR5e robotic arm, the system effectively interprets natural language commands and adjusts its actions in real-time, allowing for dynamic adaptation in collaborative environments where human proximity requires responsiveness and safety. This research represents a significant step forward in advancing human-robot collaboration by combining intuitive, language-based interaction with robust safety mechanisms.

6.1 Contributions to Research Gaps

This work addresses critical gaps in current robotic systems by combining deliberative and reactive planning capabilities. Existing systems often fall short in environments requiring adaptability to dynamic changes, especially in scenarios involving close human-robot collaboration. The hybrid planner's ability to integrate high-level task planning with real-time reactive adjustments closes this gap, ensuring that robots can operate safely and effectively in dynamic environments. Furthermore, the use of a language model like ChatGPT enhances task planning by enabling contextual understanding, which is often lacking in rule-based systems.

6.2 Implications for Human-Robot Collaboration

The findings from this project underscore the importance of integrating advanced AI models with reactive control systems to create safe and adaptive robotic systems. The hybrid framework developed in this thesis demonstrates how robots can interpret complex human commands, adapt to real-time changes in their environment, and ensure safety in collaborative settings. These capabilities have profound implications for industries such as healthcare, manufacturing, and logistics, where robots are increasingly relied upon to work alongside human operators. By bridging the gap between high-level task planning and low-level reactive control, this framework establishes a pathway for making human-robot collaboration more intuitive, efficient, and safe.

6.3 Limitations

Despite its successes, this project faced certain limitations that should be addressed in future work. There were occasional delays in detecting and responding to rapid human movements, which impacted the system's responsiveness in dynamic scenarios. Additionally, the real-world testing was limited to the global planner, while the hybrid planner was evaluated only in simulations. This constraint prevents full validation of the system's real-world performance under dynamic conditions. Furthermore, the system's scalability to more complex or populated environments remains untested, necessitating further exploration in diverse operational settings.

6.4 Future Work

To build on the progress achieved in this project, several areas for future research and development are recommended:

- **Improving Real-Time Responsiveness**: Optimization of ROS service calls, along with adjustments to MoveIt's local planner parameters could reduce latency, enabling faster and more responsive path adjustments in real-time.
- **Advanced Human Detection**: Incorporating advanced vision systems, such as LiDAR or multi-camera setups, alongside improved sensor fusion techniques, could enhance human pose detection and tracking accuracy.
- **Scaling for Complex Environments**: Future tests in larger or more populated environments, such as factory floors or clinical settings, would allow for assessment of the system's scalability and robustness when handling multiple human operators or obstacles.
- **Exploring Alternative Language Models**: Experimenting with other language models or fine-tuning prompts could enhance command interpretation accuracy and task planning capabilities, reducing potential ambiguities in complex command structures.

As robots become increasingly integrated into collaborative roles, this hybrid decision-making framework provides a foundation for future innovations. By enabling robots to operate seamlessly in environments with unpredictable human presence, the framework opens new possibilities for automation in areas where adaptability and safety are critical. This thesis lays the groundwork for developing even more sophisticated systems where robots can perform complex tasks while maintaining human trust and safety.

6.5 Final Remarks

This research contributes to the field of robotics by demonstrating a practical hybrid framework for safe and adaptable human-robot collaboration. By combining ChatGPT's advanced language understanding with reactive path planning, this project establishes a foundation for creating more intuitive and reliable robotic systems. As robotics and AI technologies continue to evolve, this work highlights the potential for these systems to seamlessly integrate into everyday human environments, enhancing productivity, safety, and accessibility. The lessons learned and innovations introduced through this research provide a strong basis for future advancements in collaborative robotics.

References

[1] Naveed: A comprehensive overview of large language models.

[2] openni2_camera wrapper, 2024. Accessed: May 01, 2024.

[3] M. Ahn et al. Do as i can, not as i say: Grounding language in robotic affordances. arXiv, Aug 2022.

[4] T. Brown et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[5] Studio Global. Ai prompt library free online. Accessed: May 01, 2024.

[6] Miguel Á. González-Santamarta. yolo_ros, February 2023. Accessed: Oct 15, 2024.

[7] H. He. Robotgpt: From chatgpt to robot intelligence, Apr 2023.

[8] A.K. Kovalev and A.I. Panov. Application of pretrained large language models in embodied artificial intelligence. *Doklady Mathematics*, 106(S1):S85–S90, Dec 2022.

[9] Y.-S. Lee and S.-B. Cho. A hybrid system of hierarchical planning of behaviour selection networks for mobile robot control. *International Journal of Advanced Robotic Systems*, 11(4):57, 2014.

[10] MoveIt Documentation. Hybrid planning concepts, 2024.

[11] Z. Mu et al. Kggpt: Empowering robots with openai's chatgpt and knowledge graph. In H. Yang et al., editors, *Intelligent Robotics and Applications*, volume 14271 of *Lecture Notes in Computer Science*, pages 340–351. Springer Nature Singapore, 2023.

[12] A. Pandya. Chatgpt-enabled davinci surgical robot prototype: advancements and limitations. *Robotics*, 12(4):97, 2023.

[13] Vasco Pires, Miguel Arroz, and Luis Custodio. Logic based hybrid decision system for a multi-robot team. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems*. Citeseer, 2004.

[14] A.-N. Sharkawy, P.N. Koustoumpardis, and N. Aspragathos. Human–robot collisions detection for safe human–robot interaction using one multi-input–output neural network. *Soft Computing*, 24(9):6687–6719, 2020.

[15] T. Takeuchi. toshiakit/matgpt, Apr 2024.

[16] W. van Dijk, S.J. Baltrusch, E. Dessers, and M.P. de Looze. The effect of human autonomy and robot work pace on perceived workload in human-robot collaborative assembly work. *Frontiers in Robotics and AI*, 10, Nov 2023.

[17] S.H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor. Chatgpt for robotics: Design principles and model abilities. *IEEE Access*, 12:55682–55696, 2024.

[18] N. Wake et al. Chatgpt empowered long-step robot control in various environments: A case application. *IEEE Access*, 2023.

[19] Y. Ye, H. You, and J. Du. Improved trust in human-robot collaboration with chatgpt. *IEEE Access*, 2023.

[20] C. Zhang, J. Chen, J. Li, Y. Peng, and Z. Mao. Large language models for human-robot interaction: A review. *Biomimetic Intelligence and Robotics*, page 100131, 2023.

[21] Haotian Zhou, Yunhan Lin, Longwu Yan, Jihong Zhu, and Huasong Min. Llm-bt: Performing robotic adaptive tasks based on large language models and behavior trees. *arXiv preprint arXiv:2404.05134*, 2024.

Appendix

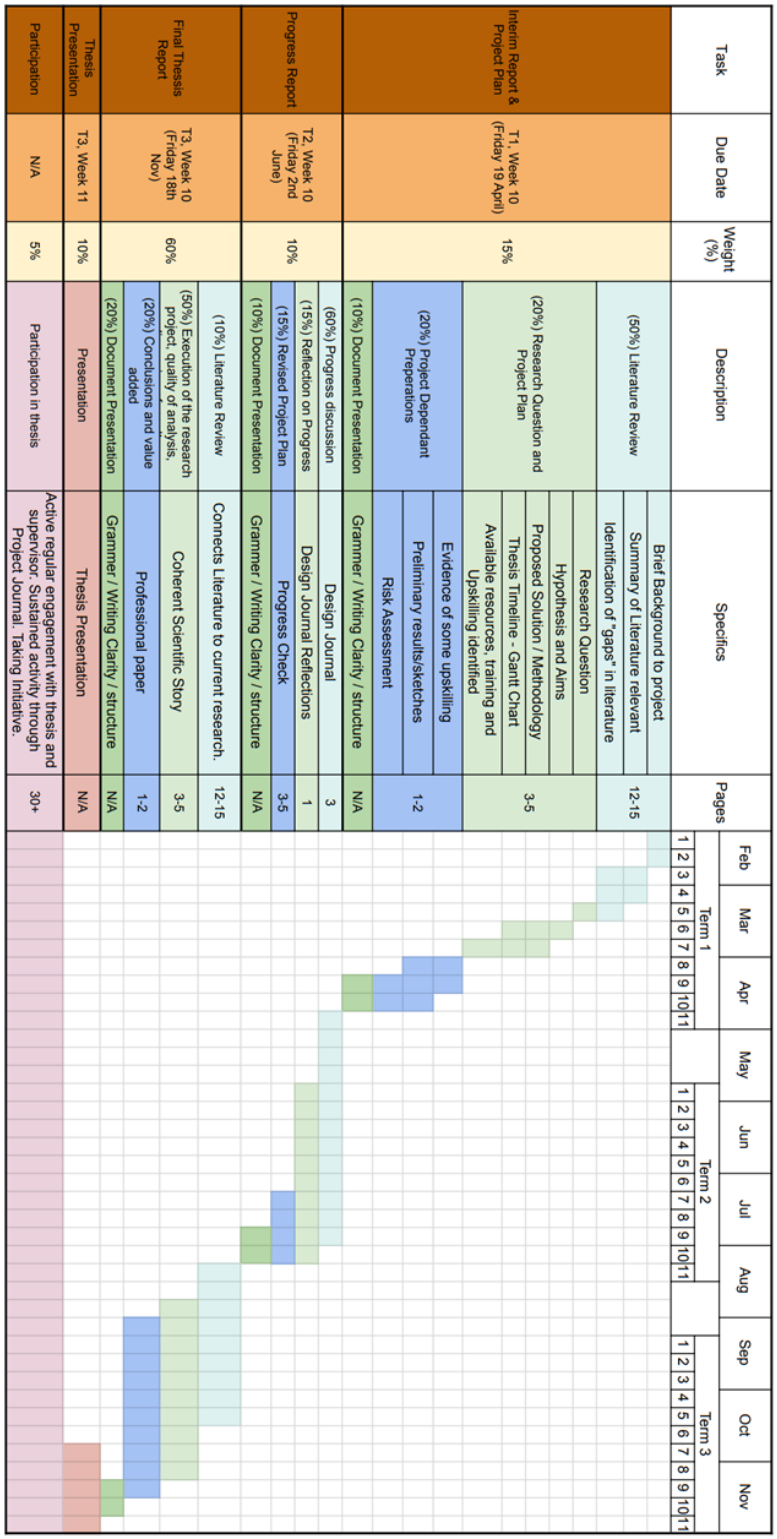


Figure 17: Gantt Chart Plan