Balancing Responsiveness, Reliability, and Flexibility in Natural-Language Robot Interaction

Timo Birr, Leonard Bärmann, Timo Weberruß, and Tamim Asfour

***Abstract*— As robotic assistants become increasingly prevalent, natural language interaction is emerging as a key enabler of effective human–robot collaboration. Although recent advances in Large Language Models (LLMs) have substantially improved open-ended language understanding, robust human–robot interaction requires more than text interpretation alone. In particular, a system must decide when an utterance warrants the robot's attention and ensure that responses are generated within acceptable time constraints. Moreover, purely LLM-based approaches often incur higher latency and are less predictable than traditional grammar-based methods when handling predefined commands. We propose a hybrid speech processing and attention management architecture that integrates low-latency command recognition with LLM-based open-vocabulary understanding. By combining the strengths of both approaches, the system achieves a balance between responsiveness and flexibility, enabling natural, efficient, and timely interaction with humanoid robots.**

I. Introduction

Instructing robots through natural language provides an intuitive and accessible mode of interaction for assistive robotic systems. However, mapping natural language utterances to concrete actions or action sequences remains a significant challenge, even with recent advances in Large Language Models (LLMs). For natural language to serve as a practical communication modality in human–robot interaction, a system must support low-latency responses while handling open-vocabulary input, inherent ambiguity, and erroneous speech recognition. Although LLMs demonstrate impressive capabilities in language understanding and task planning [1, 4], they typically require substantial computational resources and incur high latency, even for simple, low-level commands. Furthermore, due to their highly probabilistic nature, they lack predictable and explainable behavior. Before LLMs, grammar-based approaches were widely explored for natural language processing in robotics. While these methods enable efficient and predictable command execution, they heavily rely on manual engineering and don't scale well due to limited linguistic coverage, and are sensitive to paraphrasing. Motivated by the complementary strengths and limitations of these two paradigms, we propose an architecture that integrates both approaches. The system prioritizes low-latency processing through predefined grammatical structures and seamlessly falls back to an LLM when an utterance cannot be matched, enabling both responsiveness and flexibility.

The research leading to these results has received funding from the European Union's Horizon Europe programme under grant agreement No. 101070292 (HARIA), from the Carl Zeiss Foundation through the JuBot project and the German Federal Ministry of Research, Technology and Space (BMFTR) under the Robotics Institute Germany (RIG).

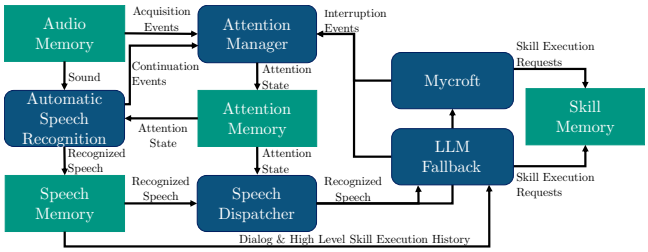

Fig. 1. Overview of the natural-language robot interaction system. Green boxes indicate memory servers, whereas blue boxes show the processing components of the system.

In addition, we introduce a dynamic and extensible speech attention mechanism that can manage multiple input cues to determine whether a given utterance is directed at the robot and should be attended to. This system, which is also shown in Figure 1, is part of the broader cognitive and memory architecture [5] of the robot software framework ArmarX [6] and has been deployed on the humanoid robot ARMAR-7.

II. Attention Management System

To determine whether an utterance is directed at the robot, we introduce an attention management system that can integrate multiple contextual cues across modalities. Rather than relying solely on speech-based wakewords – as is common in home assistants – our approach is designed for the multimodal sensing capabilities of robots, where deciding who or what to attend to is inherently a multimodal problem. The *Attention Manager* continuously tracks contextual signals and reasons over them at a higher level of abstraction to determine the robot's current focus of attention.

Attention dynamics are explicitly modeled through three event types that can be triggered by any modality: *attention acquisition, attention continuation, and attention interruption*. An *attention acquisition event* activates attention in response to explicit cues, such as a person directly addressing the robot by name. When the robot is in active attention state, it will react to user commands until it leaves the attention state due to a timeout, the start of its own speech, or an *attention interruption event.* An *attention continuation event* extends an already active attention state – for example, when the attended person continues speaking – thereby resetting the attention timeout. The attention manager publishes the robot's current attention state to the attention memory, where it can be consumed by downstream components such as speech recognition and gaze scheduling. The system is

designed to be extensible, allowing new attention cues and modalities to be integrated with minimal effort; however, in its current implementation, it only supports audio-based cues.

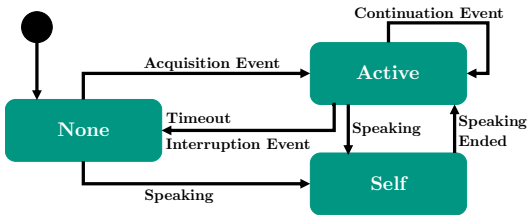

Fig. 2. Statechart for of the *Attention Manager*

III. Hybrid Processing System for Natural Language Commands

The goal of our natural language processing system is to execute an appropriate sequence of robot skills in order to fulfill commands expressed in natural language, assuming a predefined and finite skill set. In addition to physical actions, a skill may also correspond to answering a question, providing feedback, or requesting clarification from the user. Our system supports three types of natural language processing targets: i) a grammar-based processing component implemented using the *Mycroft* framework ii) an LLM-based planner for open-vocabulary command interpretation and multi-step reasoning, and iii) specialized handlers designed for short utterances in a specific context that are particularly prone to ambiguity or misinterpretation. The *speech dispatcher* dynamically selects the most appropriate processing pathway for each incoming utterance. By default, the grammar-based processing component is used to minimize latency – the LLM-based processing is only employed when the grammar-based processing cannot handle the given utterance. This design enables efficient and predictable command execution while maintaining low computational overhead, allowing the system to handle common interactions with minimal latency.

A. Grammar-Based Processing

As there is a finite set of skills available to the robot, a large subset of commands can be expressed by defining a set of natural language patterns specified via regular grammars for each possible skill. For example, the robot skill `bringObject(object, location)` can be represented by the regular expression `(Bring|Give|Fetch) me the {object} (ϵ|from the {location})`. For these commands, a fixed mapping is defined between the capturing groups of regular expressions – in this case `object` and `location` – and the corresponding skill parameters. The *Mycroft* system keeps track of all registered regular expressions, matches a given command to a regular expression, parses the parameters and executes the corresponding skill with the parsed parameters. If the command matches none of the regular expressions, a fallback is triggered that forwards the command to the LLM-based system described in the following section.

B. LLM-Based Fallback for Open-Vocabulary Commands

The LLM-based fallback is based on the dialog system introduced in our previous work [2]. This system prompts an LLM to orchestrate high-level robot behavior in a closed-loop manner. Specifically, the prompting scheme emulates a Python console environment in which the LLM observes its previous commands and their outputs and generates the next command. It can choose from the robot skill set, which is represented as an API and includes functions for perception, action, long-horizon task planning [3], and user interaction. The LLM's output is then executed within the console environment, triggering the robot's response to the user's input.

Simply using the system from [2] to process utterances that cannot be handled by the grammar-based component is insufficient. For example, consider a user saying "Move one step forward," a standard command that matches the grammar, followed by "a little further," which does not match any rule and is therefore forwarded to the LLM. However, the LLM would lack the context necessary to properly handle the request because the initial command never reached it. Therefore, we introduce an additional memory of high-level behavior invocations where the grammar-based component stores each command that it invoked. When constructing the interaction history part of the LLM's prompt, this memory is used to insert the appropriate user utterances, invoked commands, and results into the prompt. Thus, the LLM is informed of the necessary context.

Since the robot's far-field speech recognition frequently produces errors or can detect speech directed at someone else during active attention, we need a mechanism to easily ignore such inputs. Thus, we extended the API with functions to call when the recognized utterance is nonsensical or is clearly not directed at the robot. Providing these functions prevents unintended LLM responses, such as frequent clarification questions. Furthermore, when the LLM invokes the ask function to request user clarification, the speech dispatcher switches the target directly to the LLM system to prevent mapping of answers to the grammar-based system.

IV. Conclusion and Future Work

We present a scalable, low-latency system that enables natural language interaction with robots. It combines attention management with a hybrid natural language processing system that integrates a grammar-based approach with an LLM system. Currently, our implementation is limited to verbal signals for each event type. However, the system is designed to be extensible. Gestures or tactile signals could serve as attention acquisition events, and the direction of the human gaze toward the robot could trigger attention continuation events. Additionally, directing the gaze toward another human could trigger attention interruption events. Moreover, deploying smaller specialized language models might help to reduce the latency of the LLM fallback.

References

[1] Michael Ahn et al. *Do As I Can, Not As I Say: Grounding Language in Robotic Affordances*. 2022. arXiv: 2204.01691 [cs.RO].

[2] Leonard Bärmann et al. "Incremental Learning of Humanoid Robot Behavior from Natural Interaction and Large Language Models". In: *Frontiers in Robotics and AI* 11 (2024). ISSN: 2296-9144. DOI: 10.3389/frobt.2024.1455375.

[3] Timo Birr et al. "AutoGPT+P: Affordance-based Task Planning with Large Language Models". In: *Robotics Science and Systems (RSS)*. 2024.

[4] Jacky Liang et al. "Code as Policies: Language Model Programs for Embodied Control". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 9493–9500. DOI: 10.1109/ICRA48891.2023.10160591.

[5] Fabian Peller-Konrad et al. "A memory system of a robot cognitive architecture and its implementation in ArmarX". In: *Robotics and Autonomous Systems* 164 (2023), p. 104415.

[6] Nikolaus Vahrenkamp et al. "The robot software framework ArmarX." In: *it Inf. Technol.* 57.2 (2015), pp. 99–111.