Multimodal AI Agent for Orchestration of Mobile Robotic Tasks in Flexible Assembly Automation

Lucas Manassés Pinheiro de Souza[1]*, Vishnu Murugasamy[1], Alexander Moriz[1], Amon Göppert[1], Robert Schmitt[1,2]

[1] Chair for Intelligence in Quality Sensing (IQS), WZL | RWTH Aachen University, 52074 Aachen, Germany
[2] Fraunhofer Institute for Production Technology (IPT), 52074 Aachen, Germany
*Corresponding author. E-Mail address: lucas.manasses@wzl-iqs.rwth-aachen.de; Tel.:+49 241 80 20594

Abstract: Increasing product variability and shrinking batch sizes in manufacturing demand robotic systems that can be reconfigured rapidly with minimal engineering effort. While mobile manipulators provide flexibility at the hardware level, task-level specification and coordination remain a major bottleneck in assembly-oriented scenarios. This paper presents a modular multimodal AI agent that acts as a supervisory orchestration layer for mobile robotic tasks in flexible assembly automation. Following a hybrid-intelligence architecture, the agent interprets natural-language or speech instructions, grounds them in contextual and perceptual information, and coordinates navigation, perception, and manipulation via constrained Robot Operating System (ROS) 2 tool interfaces, while execution remains within established autonomy components. The framework is evaluated via quantitative analysis of end-to-end request-to-execution behavior across stepwise interaction, one-shot execution, and task reconfiguration scenarios, in simulation and selected physical validation. Results show that orchestration-related processing remains bounded relative to navigation and manipulation runtime, with execution variability dominated by perception and grounding reliability rather than agent reasoning overhead. Robust multimodal grounding substantially reduces long-tailed execution behavior in autonomous task pipelines, providing empirical insight into the runtime characteristics and limitations of agent-based task orchestration for production-relevant mobile manipulation.

Keywords:
Multimodal AI agents, Task orchestration, ROS 2, Mobile manipulation, Flexible assembly automation

1 Introduction

The manufacturing industry is undergoing a structural transformation driven by increasing product variability, shrinking batch sizes, and the need for rapid reconfiguration, further intensified by skilled labor shortages and volatile supply chains [1, 2], which forces robotic systems to operate under frequent product changes and evolving task requirements, often in close interaction with human operators [2, 3]. Mobile manipulators are a promising response because they decouple task execution from fixed layouts and enable flexible allocation of robotic resources within production environments [4, 5], making them well suited for high-mix (dis)assembly (e.g., electric vehicle battery handling or cable harness manipulation) where task sequences, grasp strategies, and motion plans must be adapted regularly [6, 7]. Despite advances in perception, navigation, and manipulation, deployment in such settings remains difficult due to the effort required to reprogram and adapt task logic as conditions change [8]; accordingly, recent work indicates that the main bottleneck has shifted from low-level capabilities to scalable task-level specification, adaptation, and orchestration, where even minor changes in (dis)assembly procedures can lead to substantial engineering effort [8, 9].

Widely adopted robotic control architectures, including finite state machines and behavior trees, have proven effective for structuring deterministic task execution, but exhibit inherent limitations when applied to flexible assembly scenarios [10]. As task complexity and variability increase, these approaches scale poorly, requiring explicit modeling of product variants, exception cases, and contingencies by domain experts [11, 10]. Moreover, they lack mechanisms to interpret semantically underspecified instructions or reason about alternative task decompositions in response to changing conditions [12, 10]. While effective for deterministic execution, classical control architectures impose rigidity that conflicts with the adaptability required in modern assembly systems [3]. To address these challenges, recent research increasingly advocates for cognitively enhanced robotic systems that combine autonomous execution with higher-level reasoning and minimal human intervention [13, 14]. Within this paradigm, hybrid intelligence emphasizes a division of roles: humans provide intent, semantic context, and supervision; AI components perform reasoning and task abstraction; and robotic software ensures safe and reliable execution [15, 16]. Advances in Large Language Models (LLMs) and Vision–Language Models (VLMs) have demonstrated strong capabilities in interpreting natural-language (NL) instructions, grounding them in visual context, and reasoning about task semantics [17, 18]. However, these models are not yet systematically integrated with established robotic frameworks to be applicable in industrial settings [17].

Although embodied AI agents combining language, vision, and action have received increasing attention [19, 20], most existing approaches focus on tabletop manipulation or highly constrained environments and rely on tightly coupled or ad-hoc architectures [2, 21]. Comparatively little work addresses task-level orchestration for mobile manipulation in assembly-oriented scenarios using modular middleware such as Robot Operating System (ROS) 2. In particular, the runtime behavior of multimodal AI agents acting as supervisory orchestration layers, bridging human intent and robotic execution, remains insufficiently characterized, with evaluations largely limited to qualitative demonstrations or task success metrics. To address this gap, this work presents a modular multimodal AI agent that acts as a supervisory orchestration layer for flexible robotic assembly tasks. The framework integrates LLMs and VLMs with a ROS 2-based mobile manipulation stack to enable NL and voice-based task specification, semantic grounding through perception, and tool-based coordination of navigation and manipulation. The approach is evaluated through a quantitative analysis of end-to-end request-to-execution behavior in both simulation and physical environments.

The remainder of this paper is organized as follows. Section 2 reviews related work on embodied AI agents and task orchestration. Section 3 presents the proposed multimodal AI agent framework and its integration with a ROS 2-based mobile manipulation stack. Section 4 describes the experimental setup and evaluation methodology. Section 5 reports and discusses the experimental results. Section 6 concludes with key findings, limitations, and directions for future work.

2 Related Work

Recent surveys on the integration of LLMs, VLMs, and multimodal foundation models into robotic systems identify a spectrum of architectural paradigms, ranging from protocol-level interfaces and language-driven command translation to tightly coupled vision–language–action models and agent-based orchestration frameworks [17, 20]. Of particular relevance to this work are *orchestration-oriented approaches*, in which an AI agent supervises and coordinates existing robotic capabilities rather than replacing them with end-to-end learned control policies, a paradigm that is especially attractive for industrial and assembly-oriented scenarios where modularity, transparency, safety, and compatibility with established robotic software stacks such as ROS 2 remain essential design requirements [9]. Within this landscape, a first class of approaches employs LLMs or VLMs primarily as semantic interfaces, translating NL instructions into symbolic task descriptions, goal specifications, or high-level action

sequences [18, 19]. While these methods demonstrate effective language understanding and multimodal grounding, they typically do not assume responsibility for task-level orchestration across perception, navigation, and manipulation, leaving execution delegated to preconfigured pipelines and limiting the agent's role to command interpretation rather than supervision of coordinated robotic behavior in dynamic environments.

Several ROS-based frameworks integrate LLM-driven agents to enable NL interaction with robots. ROS-LLM [22] maps user instructions to executable behaviors represented as state machines or behavior trees, ROSA (ROS Agent) [23] constrains LLM outputs to predefined ROS actions and services to ensure robustness and safety, and the Flexible Agent Framework for Embodied AI (RAI) [24] proposes a modular ROS 2 architecture interfacing with navigation and manipulation components. Across these frameworks, however, task execution remains bound to static or expert-defined schemas, and core mobile manipulation capabilities, such as coordinated navigation (*Nav2*), motion planning and task sequencing (*MoveIt 2* and *MoveIt Task Constructor, MTC*), *Simultaneous Localization and Mapping (SLAM*)-based localization, and 2D/3D perception-driven grasp pose estimation, are not orchestrated within a unified task-level pipeline, limiting their role to command interpretation rather than holistic supervision in assembly-oriented scenarios.

Beyond ROS-centric solutions, orchestration-oriented agentic systems have been explored primarily in non-industrial contexts. AutoRT [25] applies LLM-based supervision to large robot fleets for autonomous data collection, while the LABOR Agent [26] investigates coordination of manipulation skill libraries for bimanual tasks, but these approaches are evaluated in constrained or non-mobile settings and are not designed around ROS 2-native modular autonomy stacks. In parallel, tightly coupled vision–language–action models such as RT-2 [27] unify perception, reasoning, and execution within a single learned model; although powerful, such end-to-end approaches complicate integration with modular robotic architectures, reduce transparency and debuggability, and pose challenges for safety assurance and certification in industrial environments [28]. Nevertheless, such models can be integrated as specialized perception or grounding components, provided that task-level coordination and execution remain governed by deterministic, ROS 2-native software.

Table 1: Comparison of related frameworks with the proposed approach. Legend: ● supported, ◐ partially supported/limited, ✗ not addressed. ROS 2-Components refers to the integration of motion planning (*MoveIt 2, MTC*), navigation (*Nav2*), *SLAM*, perception, and pose estimation for grasping.

Framework	Agent Orchestration	Mobile Manipulation	ROS 2 Native	ROS 2 Components	Runtime Behavior Evaluated	Assembly Automation (Production)	Hybrid Intelligence
ROS-LLM [22]	◐	✗	●	◐	●	◐	●
ROSA [23]	◐	✗	●	◐	◐	✗	◐
RAI [24]	●	✗	●	◐	◐	✗	◐
AutoRT [25]	✗	◐	✗	✗	●	✗	◐
LABOR Agent [26]	◐	✗	✗	✗	●	✗	◐
Proposed approach	●	●	●	●	●	●	●

Across the reviewed literature, two recurring limitations emerge. First, most existing frameworks focus on tabletop manipulation or highly constrained environments, with limited consideration for mobile manipulation scenarios that require simultaneous coordinated navigation, perception, and manipulation. Second, evaluations predominantly emphasize task success or qualitative demonstrations, while the runtime behavior of AI-agent-based orchestration pipelines, including interaction overhead, request-to-execution latency, and responsiveness to high-level human input, remains insufficiently characterized. In contrast, the proposed approach explicitly integrates motion planning and task sequencing (*MoveIt* 2, *MTC*), autonomous navigation (*Nav2*), *SLAM*-based localization, perception, and grasp pose estimation

into a unified orchestration pipeline. Rather than benchmarking isolated capabilities, this work investigates the runtime behavior of a multimodal AI agent acting as a supervisory layer, providing empirical insight into how high-level human intent is translated into coordinated robotic actions in production-relevant scenarios. A comparative overview of representative frameworks and the proposed approach is summarized in Table 1.

3 Multimodal AI Agent for Hybrid-Intelligent Task Orchestration

The proposed framework introduces a multimodal AI agent that acts as a supervisory orchestration layer for mobile manipulation in flexible assembly scenarios. The architecture follows a hybrid-intelligence split: the operator provides intent and supervision; the agent interprets instructions and coordinates task execution; and established ROS 2 autonomy components perform navigation and manipulation. This design targets frequent task changes and semantically underspecified requests while preserving execution within the underlying robotics stack.

3.1 Framework Overview

High-level user inputs are provided as text or speech and interpreted by the agent in conjunction with contextual state (e.g., robot pose, semantic waypoints) and perception outputs. The agent operates exclusively at the task level by selecting and parameterizing tools from ROS 2 subcomponents, while motion generation, collision checking, and control remain within the underlying autonomy stack. Tool execution feedback is returned to the agent to support iterative orchestration. Figure 1 summarizes the architecture.

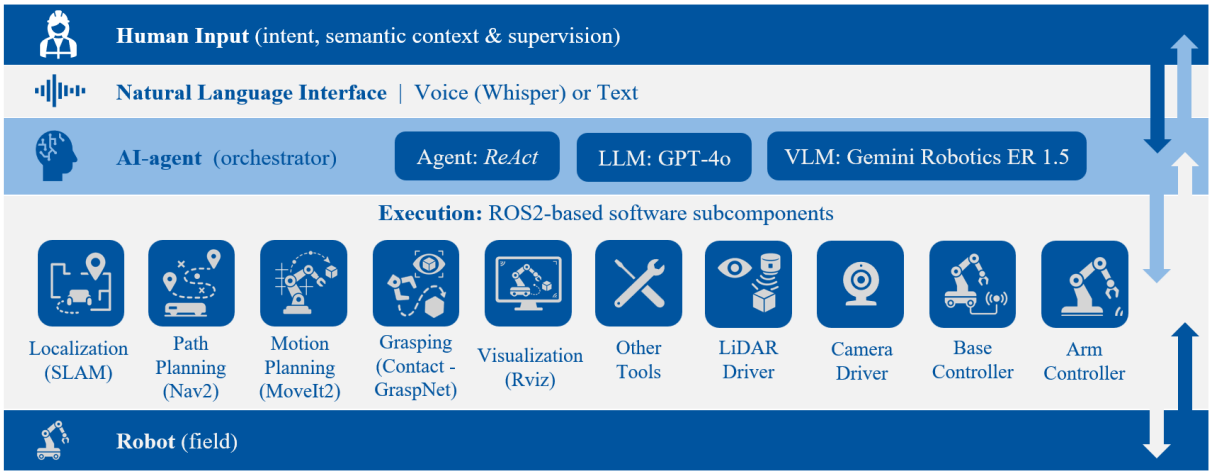

Figure 1: Layered architecture of the multimodal AI-agent orchestration layer integrated with ROS 2 modules for mobile manipulation. High-level inputs (text/speech) are interpreted by the agent powered by the LLM & VLM models. The agent then invokes modular ROS 2 tools (*SLAM*, *Nav2*, *Contact-GraspNet*, *MoveIt 2/MTC*). Execution feedback closes the loop for iterative orchestration.

3.2 Multimodal Agent and Tool-Based Orchestration

The orchestration logic follows an iterative reason–act–observe (ReAct-style) pattern, in which high-level instructions are decomposed into short-horizon tool invocations and revised based on runtime feedback. The agent combines NL or speech input with structured robot state and optional visual context to support task interpretation and grounding. In the presented implementation, semantic interpretation and task decomposition are handled by an LLM instantiated using *GPT-4o API* [29], which parses NL or transcribed speech instructions to extract task intent, entities, and ordering constraints. For perception-driven grounding, the framework also employed Grounded SAM (GSAM) [30] for segmentation of the target object, later fully replaced by vision–language capabilities provided by *Gemini Robotics ER 1.5*

API [31], which supports perception-driven grounding, for example by assisting object identification and semantic referencing during grasp pose estimation. Outputs from both models are constrained through predefined tool schemas and prompt design, ensuring that only valid robot capabilities can be invoked and that ambiguous instructions are handled conservatively.

Each robotic capability is exposed to the agent through a narrow tool interface aligned with ROS 2 actions, services, and topics, including waypoint management, mapping and localization (SLAM), autonomous navigation (*Nav2*), perception and object pose estimation (e.g., *Contact-GraspNet*), motion planning and task execution (*MoveIt 2* and *MTC*), and bounded teleoperation primitives for base and arm control. This abstraction decouples the orchestration logic from the internal implementation of individual modules, allowing components such as *SLAM* backends, perception models, or grasp generators to be replaced or upgraded without modifying the agent's reasoning or task sequencing logic. Each tool invocation returns structured execution feedback, enabling the agent to detect failures and potentially react through retries, or operator interaction. To support reusable task references, the framework includes a lightweight semantic memory mechanism in which operators label base and arm waypoints during setup (e.g., "pick-up station", "assembly station", "scan pose", "pre-grasp"). Waypoints can be created via teleoperation or incremental motion primitives and are stored as callable parameters for subsequent autonomous execution. Semantic memory enables hybrid task execution: operators provide sparse semantic structure once, and the agent resolves later references (e.g., "go to the assembly station") into explicit goals for navigation or manipulation tools. This reduces reconfiguration effort when stations, targets, or task order change, while keeping execution within deterministic autonomy modules.

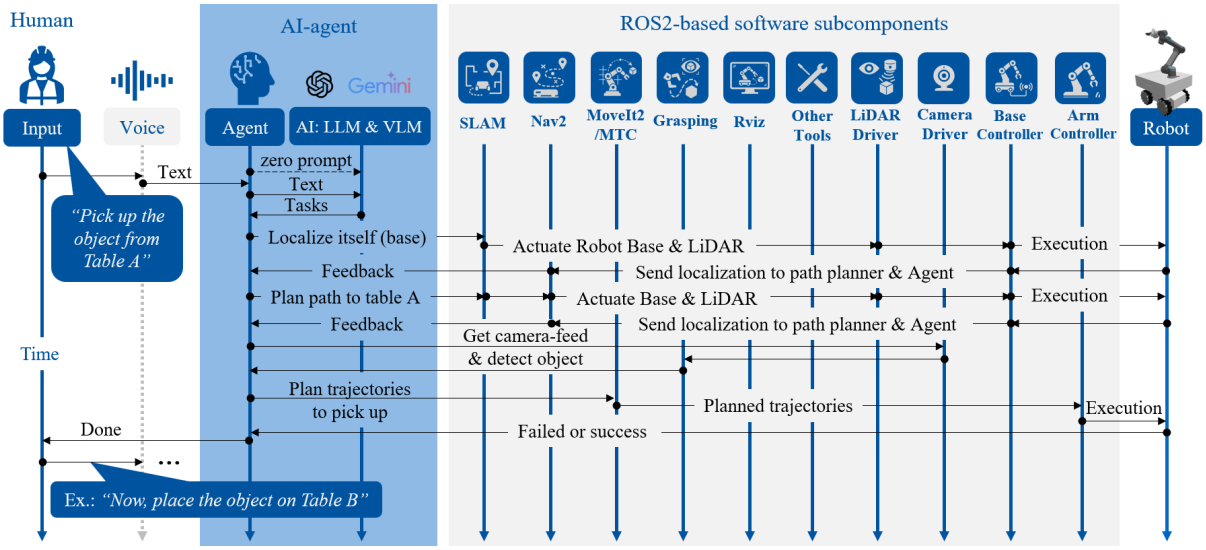

Figure 2: Exemplary workflow for hybrid-intelligent orchestration in an assembly-oriented pick-and-place task. The operator defines semantic stations and poses, while the agent decomposes a high-level instruction into navigation (*Nav2*), perception, and manipulation (*MoveIt 2/MTC*) tool calls with feedback-driven sequencing.

3.3 Exemplary Orchestration Workflow

Figure 2 illustrates an assembly-oriented pick-and-place workflow executed from a single high-level instruction, e.g., "Go to the pick-up station, pick the gear, and place it on the assembly station." The agent parses the request to identify referenced locations and objects and resolves them via semantic memory (labeled positions). If required semantic labels are missing, the agent requests operator input; otherwise, it autonomously sequences navigation, perception, and manipulation. The agent issues navigation goals to reach the pick-up and assembly stations, triggers perception to estimate grasp poses for the target object and invokes motion planning and execution for pick and place using *MoveIt 2* /

MTC. Execution feedback from each tool is monitored to support retries, or operator intervention when necessary. In Figure 3 the Hybrid-intelligent orchestration workflow is shown in the simulation.

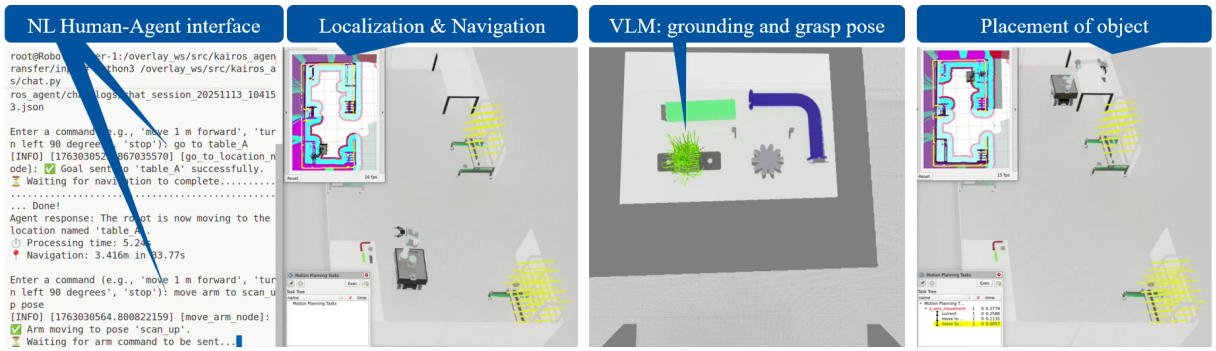

Figure 3: Hybrid-intelligent orchestration workflow in simulation. Left: NL human–agent interface for defining semantic goals, pose labeling, and teleoperation, shown with real-time *SLAM* and *Nav2*-based navigation in simulation environment as the robot moves to *Table A* to pick the target object (T-connector). Middle: VLM-based perception-driven grounding and grasp pose generation from the arm camera. Right: Autonomous placement of the object on *Table B* as requested by human input.

4 Experimental Setup

The evaluation focuses on end-to-end request-to-execution behavior and orchestration overhead under assembly-oriented conditions. Experiments were conducted in NVIDIA Isaac Sim to ensure repeatability, with one selected validation scenario being performed in a physical setup, as shown in Figure 4 (left). For the simulation environment, Figure 4 (middle), a simulated mobile manipulator, Figure 4 (right), based on the RB-Kairos platform, equipped with a 6-DoF industrial arm, an RGB-D camera, and LiDAR sensors for navigation. The simulated setup exposes the same ROS 2 interfaces as the physical platform, allowing the orchestration pipeline to operate with minimal changes. The software stack includes *SLAM*, *Nav2*, *Contact-GraspNet*, *MoveIt 2* and *MTC*. The multimodal AI agent framework runs as a supervisory layer, coordinating these components through structured tool invocations.

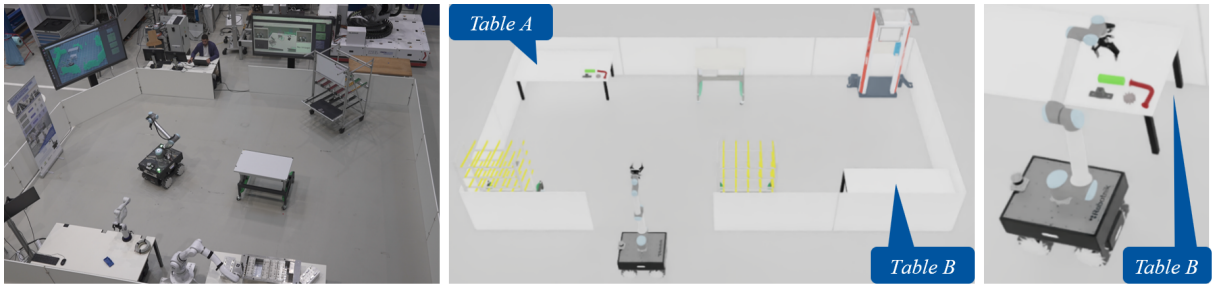

Figure 4: Left: physical setup at WZL RWTH. Middle: simulated environment in NVIDIA Isaac Sim. Right: simulated mobile manipulator RB-Kairos, equipped with a 6-DoF industrial arm, simulated RGB-D camera, and simulated LiDAR sensors for navigation and manipulation.

Orchestration-layer behavior is evaluated using (i*) processing time* (request-to-plan), measured as the time from user request to the agent producing an executable tool sequence, (ii) *execution time* (request-to-execution), measured as the time from request to completion of robot actions, and (iii) *orchestration share*, computed as $\text{Proc}/(\text{Proc} + \text{Exec}) \times 100$. In addition, execution variability is reported qualitatively (Low/Med/High) based on observed spread and long-tailed runtimes, and notable agent/runtime events (e.g., retries, restarts, drift) are logged as annotations. For the physical validation (Scenario 4), timing is additionally reported per subsystem (navigation and arm motion) to reflect real-world execution effects.

Scenario 1 (S1) Stepwise Interaction with Explicit Subtasks, where the user issues individual high-level commands for discrete subtasks (e.g., navigation, object detection, pick), but still in a logical sequence. Commands are provided via text and voice, and the agent executes each step separately. This scenario serves as a baseline for evaluating interaction overhead and agent responsiveness, isolating latency from instruction interpretation, tool invocation, and feedback handling under controlled conditions.

Scenario 2 (S2) One-Shot Task Execution (Pipeline Orchestration), where the user provides a single high-level instruction (one-shot task execution) specifying a complete assembly task (navigate–pick–place). The agent autonomously decomposes and sequences navigation, perception, and manipulation actions without further user input. This scenario assesses task-level decomposition and orchestration behavior, capturing the cumulative overhead of semantic interpretation, tool selection, and feedback-driven sequencing.

Scenario 3 (S3) Environmental Change and Task Reconfiguration, where changes are introduced to the environment or task structure (e.g., modified object locations or station assignments), as shown in Figure 5. The user issues high-level instruction similar to Scenario 2, while the agent adapts execution based on updated context and stored semantic labels. This scenario evaluates the agent's ability to reuse semantic memory, request clarification when needed, and re-sequence actions without manual task-graph reprogramming.

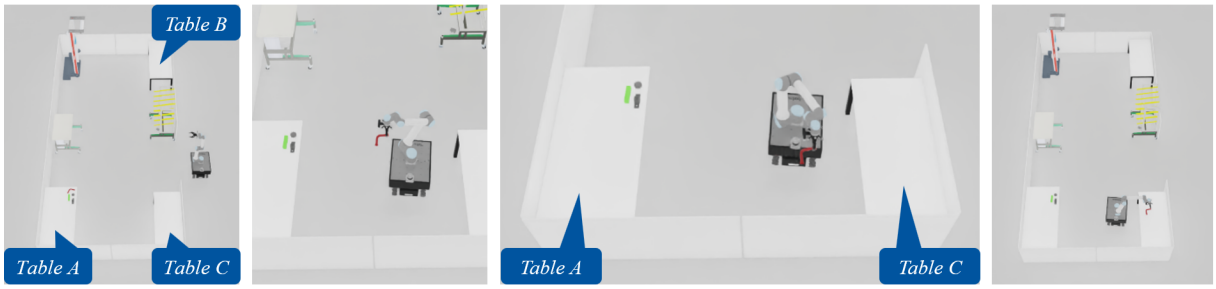

Figure 5: Evaluation of Scenario 3. An additional placement station is introduced into the environment (e.g., *Table C*), requiring the agent to resolve updated semantic labels and adapt task sequencing.

Scenario 4 (S4) Physical Validation and Transferability, where transferability of the stepwise interaction (Scenario 1) is assessed through deployment on a physical setup, as shown in Figure 6. The physical platform mirrors the simulated configuration in terms of software architecture and ROS 2 interfaces. This scenario evaluates integration of Nav2-based navigation and arm control for executing tasks at labeled semantic positions. While quantitative analysis focuses on simulation, physical execution confirms consistent orchestration behavior under real sensing and actuation conditions, with only minor platform-specific adaptations required.

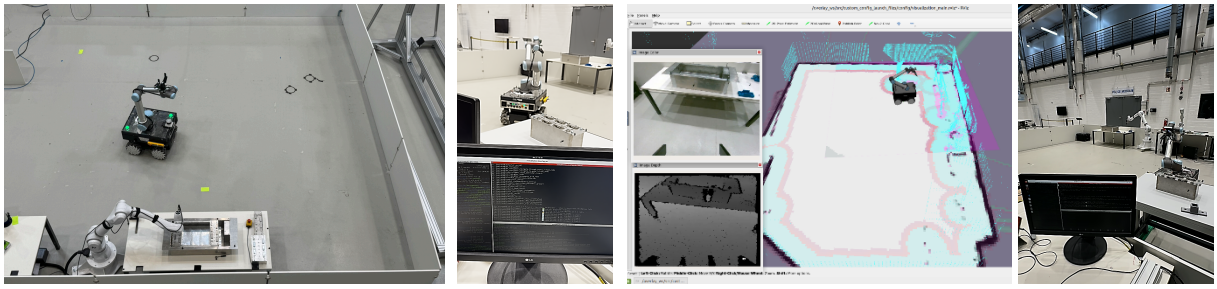

Figure 6: Physical validation of Scenario (4). Left: Mobile manipulator in the physical evaluation setup. Middle left: Onboard execution of the multimodal AI agent with NL interface. Middle right: Real-time RViz visualization of the ROS 2 stack during navigation and manipulation. Right: Robot reaching the labeled placement station (Table B) with the NL interface visible.

5 Results and Discussion

This section presents the experimental results for each of the previously defined scenarios. Table 2 summarizes the end-to-end behavior of the proposed orchestration layer across all four scenarios and interaction/grounding configurations. Overall, orchestration processing remains bounded relative to navigation and manipulation runtime, and variability is primarily driven by perception/grounding reliability and physical execution effects rather than agent reasoning.

Table 2: Aggregated orchestration performance across scenarios. Plan [s]: request-to-plan latency (user request → executable tool sequence, including parsing and tool selection). Exec [s]: request-to-execution latency (user request → completion of robot actions). Orch [%]: orchestration share of total runtime, computed as Plan/(Plan+Exec). Var: qualitative execution-time variability (L = Low, M = Medium, H = High) based on observed spread and long-tailed runtimes. ↑ indicates increased planning effort relative to S2 due to semantic context resolution (Table C).

Scenario	Interact.	Ground.	n	Plan [s]	Exec [s]	Orch [%]	Var	Note
S1 (stepwise)	Text	-	12	27.6 ± 5.2	139.5 ± 24.2	16.5	L	Baseline
	Voice	-	12	39.2 ± 13.6	379.5 ± 135.6	9.4	M	Speech overhead
S2 (one-shot)	Text	GSAM	13	-	308.9 ± 214.6	~10	H	Grasp retries
	Text	Gemini	15	-	155.3 ± 22.8	~12	L	Stable grounding
	Voice	Gemini	22	-	159.8 ± 24.7	~12	L	No added instability
S3 (reconfig.)	Text	Gemini	10	↑ vs S2	↑ / ~stable vs S2	~12–14	L	Context (Table C)
S4 (physical)	Text (stepwise)	Gemini	10	3.05 ± 1.16	73.21 ± 29.18	-	M–H	Nav: Retries/drift/restart
			8	1.84 ± 0.79	4.01 ± 1.81	-	L	Arm: Retries/restart

Across 24 stepwise runs (S1), request-to-plan behavior is stable for both text and voice input. Voice interaction increases processing time due to speech transcription and clarification handling, but orchestration remains a minority of total task duration; overall runtime is dominated by navigation and manipulation, with execution variance mainly reflecting differences in the number and type of issued subtasks rather than orchestration instability.

For one-shot execution (S2, n=50), the agent incurs planning/reasoning overhead only once per task, and performance is dominated by grounding reliability. Chat + GSAM shows high variance and long-tailed runtimes due to repeated grasp attempts, whereas Gemini-based grounding yields substantially more stable execution with similar behavior for text and voice interaction, indicating that robust multimodal grounding, not interaction modality, drives predictability in autonomous pipeline orchestration.

Under environmental change and task reconfiguration (S3, n=10), introducing a new semantic station (*Table C*) increases planning-time overhead due to context resolution, while execution remains dominated by navigation and manipulation. No increase in execution variance or grasp retries is observed, suggesting that moderate environment/task changes can be accommodated through semantic memory without manual task-graph modification.

Physical validation (S4, n=10) confirms transferability of the stepwise workflow when ROS 2 interfaces are preserved. Processing latency remains low (navigation: 3.05 ± 1.16 s; arm: 1.84 ± 0.79 s), while runtime variability is dominated by real-world execution effects (navigation: 73.21 ± 29.18 s; arm: 4.01 ± 1.81 s) and occasional operational interventions (e.g., repeated commands, script restart, minor pose drift), indicating that sim-to-real deviations arise primarily outside the orchestration layer.

6 Conclusion, Limitations, and Outlook

This paper presented a modular multimodal AI agent that acts as a supervisory orchestration layer for mobile robotic tasks in flexible assembly automation. The agent interprets natural-language or speech

instructions and coordinates navigation, perception, and manipulation through constrained ROS 2 tool interfaces, while low-level execution remains within the autonomy stack. Across simulation and physical trials, processing time remained bounded relative to overall task duration, and end-to-end behavior was dominated by navigation and manipulation runtime rather than agent reasoning.

The evaluation highlights three main findings. First, under stepwise interaction (S1), processing time is stable for text and voice input; voice increases processing latency but does not change the dominant role of execution time. Second, in one-shot execution (S2), runtime predictability is driven primarily by grounding reliability: GSAM leads to long-tailed execution due to grasp retries, whereas Gemini-based grounding yields substantially more stable execution for both text and voice commands. Third, introducing a new semantic station (S3) increases planning-time effort due to context resolution, but does not increase execution variability, indicating that lightweight semantic memory supports moderate task reconfiguration without manual task-graph modification. Physical validation (S4) confirms transferability when ROS 2 interfaces are preserved, while remaining variability is largely attributable to real-world sensing and execution effects.

Limitations include a mainly simulation-based evaluation, a restricted physical benchmark, and the absence of controlled baselines against expert-engineered task graphs or classical planning approaches. Agent behavior (e.g., retries, interventions) was logged qualitatively rather than systematically counted, and request-to-plan was not reported for all one-shot configurations. Future work will extend physical benchmarking to full end-to-end pick-and-place pipelines under controlled disturbance factors, and will introduce explicit baselines to quantify reprogramming effort and runtime trade-offs against classical task representations. Robustness can be improved by adding failure-aware recovery strategies, tighter monitoring of tool-level outcomes, and more systematic logging of clarification and retry events. Beyond short-horizon tasks, the framework should be scaled toward longer-horizon and contact-rich (dis)assembly by integrating stronger safety layers, uncertainty-aware execution, and richer memory mechanisms (e.g., episodic traces) to support adaptation across runs.

Acknowledgements

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and from the German Federal Ministry of Research, Technology and Space (BMFTR), the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) and the Hessian Ministry of Science and Research, Arts and Culture (HMWK) under grant agreement No 101250682.

References

[1] F. Bermpohl, S. F. Schäfer, A. Hohmann und R. Daub, „Short-cycled factory planning – motivation and existing challenges," *Procedia CIRP,* p. 1708–1713, 2024.

[2] A. Keshvarparast, D. Battini, O. Battaia und A. Pirayesh, „Collaborative robots in manufacturing and assembly systems: literature review and future research agenda," *Journal of Intelligent Manufacturing,* Bd. 35, pp. 2065-2118, 2023.

[3] S. Asif, M. Bueno, P. Ferreira, P. Anandan, Z. Zhang, Y. Yao, G. Ragunathan, L. Tinkler, M. Sotoodeh Bahraini, N. Lohse, P. Webb, W. Hutabarat und A. Tiwari, „Rapid and automated configuration of robot manufacturing cells," *Robotics and Computer-Integrated Manufacturing,* Bd. 92, p. 102862, 2025.

[4] G. Hüttemann, Modellbasierte a priori Bewertung von linienlosen mobilen Montagesystemen, Aachen: RWTH Aachen University, 2020.

[5] L. Bergs, G. Tréheux, L. Schäper, L. Manassés Pinheiro de Souza, S. Münker, A. Göppert und R. H. Schmitt, „Edge Computing Framework for Enhanced Robotic Adaptivity in Line-Less Mobile Assembly Systems," *Procedia CIRP,* Bd. 127, pp. 26-31, 2024.

[6] Y. Zang, M. Qu, D. T. Pham, R. Dixon, F. Goli, Y. Zhang und Y. Wang, „Robotic disassembly of electric vehicle batteries: Technologies and opportunities," *Computers & Industrial Engineering,* Bd. 198, p. 110727, 2024.

[7] D. Gebauer, J. Dirr und G. Reinhart, „Concept for Robot-Based Cable Assembly Regarding Industrial Production," in *Annals of Scientific Society for Assembly, Handling and Industrial Robotics 2021*, Cham, 2022.

[8] Z. Jin, R. M. Marian und J. S. Chahl, „Achieving batch-size-of-one production model in robot flexible assembly cells," *The International Journal of Advanced Manufacturing Technology,* Bd. 126, Nr. 5, pp. 2097-2116, 2023.

[9] T. Wiese, J. Abicht, A. Dietz und T. Bauernhansl, „An evaluation framework for assembly planning approaches in robotics and AI," *Frontiers in Robotics and AI,* Bd. 9, p. 1014476, 2022.

[10] R. Ghzouli, T. Berger, E. B. Johnsen, A. Wasowski und S. Dragule, „Behavior trees and state machines in robotics applications," *IEEE Transactions on Software Engineering,* Bd. 49, Nr. 9, pp. 4243-4267, September 2023.

[11] J. Cloete, W. X. Merkt und I. Havoutis, „Adaptive manipulation using behavior trees," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[12] R. Liu, G. Wan, M. Jiang, H. Chen und P. Zeng, „Autonomous robot task execution in flexible manufacturing: integrating PDDL and behavior trees in ARIAC 2023," *Biomimetics,* Bd. 9, Nr. 10, p. 612, 2024.

[13] M. Soori, B. Arezoo und R. Dastres, „Artificial intelligence, machine learning and deep learning in advanced robotics, a review," *Cognitive Robotics,* Bd. 3, pp. 54-70, 2023.

[14] J. T. Licardo, M. Domjan und T. Orehovački, „Intelligent robotics—A systematic review of emerging technologies and trends," *Electronics,* Bd. 13, Nr. 3, p. 542, 2024.

[15] C. R. Sauer und P. Burggräf, „Hybrid intelligence – systematic approach and framework to determine the level of Human-AI collaboration for production management use cases," *Production Engineering,* Bd. 19, Nr. 3, pp. 525-541, 2025.

[16] H. Zhang, Y. Zhang, Z. Wang, S. Zhang, H. Li und M. Chen, „A novel knowledge-driven flexible human–robot hybrid disassembly line and its key technologies for electric vehicle batteries," *Journal of Manufacturing Systems,* Bd. 68, pp. 338-353, 2023.

[17] J. Fan, Y. Yin, T. Wang, W. Dong, P. Zheng und L. Wang, „Vision-language model-based human-robot collaboration for smart manufacturing: A state-of-the-art survey," *Frontiers of Engineering Management,* Bd. 12, Nr. 1, p. 177, 2025.

[18] H. Jeong, H. Lee, C. Kim und S. Shin, „A Survey of Robot Intelligence with Large Language Models," *Applied Sciences,* Bd. 14, Nr. 19, p. Article Number: 8868, 2024.

[19] W. Huang, P. Abbeel, D. Pathak und I. Mordatch, „Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents," arXiv, 2022.

[20] Y. Ren, Y. Liu, T. Ji und X. Xu, „AI agents and agentic AI–navigating a plethora of concepts for future manufacturing," *Journal of Manufacturing Systems,* Bd. 83, pp. 126-133, 2025.

[21] H. Guo, F. Wu, Y. Qin, R. Li, K. Li und K. Li, „Recent Trends in Task and Motion Planning for Robotics: A Survey," *ACM Computing Surveys,* Bd. 55, Nr. 13s, p. 289, 2023.

[22] C. E. Mower, Y. Wan, H. Yu, A. Grosnit, J. Gonzalez-Billandon, M. Zimmer, J. Wang, X. Zhang, Y. Zhao, A. Zhai, P. Liu, D. Palenicek, D. Tateo, C. Cadena, M. Hutter, J. Peters, G. Tian und Y, „ROS-LLM: A ROS framework for embodied AI with task feedback and structured reasoning," 2024.

[23] R. Royce, M. Kaufmann, J. Becktor, S. Moon, K. Carpenter, K. Pak, A. Towler, R. Thakker und S. Khattak, „Enabling Novel Mission Operations and Interactions with ROSA: The Robot Operating System Agent," arXiv, 2024.

[24] K. Rachwał, M. Majek, B. Boczek, K. Dąbrowski, P. Liberadzki, A. Dąbrowski und M. Ganzha, „RAI: Flexible Agent Framework for Embodied AI," in *Highlights in Practical Applications of Agents, Multi-Agent Systems and Computational Social Science. The PAAMS Collection*, Cham, 2025.

[25] M. Ahn, D. Dwibedi, C. Finn, M. Gonzalez Arenas, K. Gopalakrishnan, K. Hausman, B. Ichter, A. Irpan, N. Joshi, R. Julian, S. Kirmani, I. Leal, E. Lee, S. Levine, Y. Lu und S. Maddineni, „AutoRT: Embodied Foundation Models for Large Scale Orchestration of Robotic Agents," 2024.

[26] K. Chu, X. Zhao, C. Weber, M. Li, W. Lu und S. Wermter, „Large Language Models for Orchestrating Bimanual Robots," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, Nancy, France, 2024.

[27] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. Gonzalez Arenas, K. Gopalakrishnan und K. Han, „RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control," 2023.

[28] A. S. Adebayo, O. O. Ajayi und N. Chukwurah, „Explainable AI in Robotics: A Critical Review and Implementation Strategies for Transparent Decision-Making," *Journal of Frontiers in Multidisciplinary Research,* Bd. 05, Nr. 01, p. 26–32, 2024.

[29] OpenAI, „GPT-4o System Card," 2024.

[30] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang und L. Zhang, „Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks," 2024.

[31] Gemini Robotics, „Gemini Robotics 1.5: Pushing the Frontier of Generalist Robots with Advanced Embodied Reasoning, Thinking, and Motion Transfer," 2025.