

13th CIRP Global Web Conference (CIRPe 2025)

Embodied AI Agent Framework for (Re)Programming Robotic Tasks in Flexible Assembly using LLMs and VLMs

Lucas Manassés Pinheiro de Souza^{a,*}, Abhishek Deshmukh^a, Kedar Rajpathak^a, Alexander Moriz^a, Amon Göppert^a, Robert H. Schmitt^{a,b}

^a Laboratory for Machine Tools and Production Engineering (WZL), RWTH Aachen University, Campus Boulevard 30, 52074 Aachen, Germany

^b Fraunhofer Institute for Production Technology (IPT), 52074 Aachen, Germany

* Corresponding author. Tel.: +49-241-802-0594 ; fax: +49-241-802-2293. E-mail address: lucas.manasses@wzl-iqs.rwth-aachen.de

Abstract

The increasing product variability and skilled labor shortages in manufacturing intensify the need for more flexible and adaptive automation solutions, particularly in assembly systems. However, existing robotic automation typically requires expert-level programming, preventing adaptability to new tasks or product variants. This work presents a modular embodied AI agent framework that enables non-programming operators to implement and reconfigure robotic tasks through intuitive natural language (NL) commands in text or voice format. By integrating open-source Large Language Models (LLMs) and Visual Language Models (VLMs) with a Robot Operating System 2 (ROS2)-based stack, the agent translates user instructions into perception, grasp generation, and motion planning actions. The framework is evaluated in real-world experiments across four scenarios: low-level motion control, vision-guided cable grasping, visual scene feedback, and task-cycle recording with replay for scalable execution. Results show 100% command parsing accuracy for basic motions, reliable visual feedback, and 70% success in dual-cable manipulation, with failures mainly from trajectory planning or self-collisions. VLM inference latency proved highly hardware-dependent, with near real-time performance on high-end GPUs but significant slowdowns on consumer devices, highlighting deployment challenges. These findings show the potential of embodied AI agents to bridge NL interaction with robotic execution, lowering barriers to deploying adaptive, operator-friendly robotic systems in dynamic manufacturing.

© 2025 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<https://creativecommons.org/licenses/by-nc-nd/4.0>)

Peer review under the responsibility of the scientific committee of the CIRPe 2025

Keywords: Embodied AI; AI Agent; LLMs/VLMs; ROS 2; Flexible Assembly Automation

1. Introduction

The manufacturing industry is undergoing a paradigm shift driven by demands for shorter production cycles and greater customization [1]. Automation has emerged as a key enabler, helping address persistent challenges such as skilled labor shortages [2], increasing product diversity [3], and volatile supply chains [4]. In particular, flexible robotic assembly systems have gained traction [5], as they offer the ability to handle high-mix production scenarios with dynamic task requirements.

Yet, enabling such adaptability remains a key challenge. Product variants, for example those in electric vehicle battery designs [6] or cable harness assembly [7], often demand frequent updates to task sequencing, grasp strategies, and motion planning. Even minor changes can necessitate time-consuming

reprogramming [8]. Existing control architectures based on state machines or behavior trees are typically brittle and hard to scale [9, 10, 11], requiring expert input for adaptation [12, 13]. This results in high integration overhead and limits deployment in dynamic manufacturing contexts [14, 15].

To overcome these limitations, robotic systems must evolve toward cognitive autonomy, becoming capable of perceiving, reasoning, and adapting in real-time with minimal human intervention [16, 17, 18]. Recent advances in Large Language Models (LLMs) and Vision-Language Models (VLMs) open new avenues for intuitive, multimodal human-robot interaction. These models can parse high-level commands (e.g., “pick up the cable”) and ground them in scene context via visual reasoning and task knowledge [19, 20].

Building on these capabilities, embodied AI agents, which are robotic systems equipped with semantic understanding of tasks, have emerged as a promising paradigm for high-level autonomy in robotics [21, 22]. This paper presents a modular em-

bodied AI agent framework for reprogrammable robotic assembly, integrating open-source LLMs, VLMs, and a Robot Operating System 2 (ROS2)-based architecture. The framework enables non-expert users to issue natural language (NL) or voice commands, which are interpreted and grounded into robot behaviors through modular components for perception, grasping, and motion planning, thereby supporting rapid task reconfiguration in dynamic assembly scenarios.

The remainder of this paper is structured as follows. Section 2 reviews recent work on integrating LLMs, VLMs, and AI-agent architectures for robotic applications, highlighting current limitations. Section 3 introduces the architecture of the embodied AI agent, including its modular ROS2-based software (SW) stack, integration with open-source LLMs/VLMs, and the grasping component. Section 4 describes the experimental setup and test scenarios, while Section 5 discusses the results of real-world evaluations. Section 6 concludes with key findings and future research directions.

2. Related Work

Recent frameworks have integrated LLMs and VLMs into robotic workflows to enable high-level task specification and multimodal grounding [19, 20, 22]. While these efforts show strong potential for minimal human innervation in reprogramming robots, most remain limited to research settings and lack integration with modular, real-time control ROS-stacks suitable for dynamic assembly automation.

The ROS-LLM [23] provides a ROS-based interface for programming robots using NL, supporting structured feedback and task execution via behavior trees and state machines. While it incorporates imitation learning and user feedback to refine behaviors, it does not introduce an autonomous agent capable of dynamic decision-making or tool orchestration; instead, it relies on predefined action sequences and static control structures. The ROS Agent (ROSA) [24] proposes a modular agent that maps NL to ROS 2 actions using structured representations. Although it enables multimodal interaction and supports safety constraints, task orchestration remains manually defined, requiring expert input to extend or adapt the system. Furthermore, ROSA does not integrate widely adopted robotics libraries such as MoveIt2 or the MoveIt Task Constructor (MTC), limiting its capacity for trajectory planning, inverse kinematics, or high-level task sequencing. PickGPT [25], a commercial system designed for logistics picking, focuses on closed-loop object manipulation. Although optimized for performance, it is not structured as an autonomous agent and lacks general-purpose planning, modularity, or extensibility due to its proprietary design. The Flexible Agent Framework for Embodied AI (RAI) [26] introduces an agent architecture with episodic memory and hybrid symbolic-LLM planning, supporting multi-agent setups and general reasoning. While RAI extends ROS2-based SW components and supports high-level prompts via LLMs, it lacks integration with advanced tools like the MTC and does not offer modular action primitives for task sequencing. It also lacks a dedicated mechanism for decomposing NL instructions into se-

mantically grounded actions, relying instead on prompt-based reasoning using cloud-based LLMs and VLMs. These choices highlight flexibility but leave open directions for enhancing autonomy, task generalization, and on-device decision-making in more complex assembly scenarios. A comparative summary of these frameworks is presented in Table 1.

Additionally, grasp generation plays a critical role in translating high-level goals into executable robotic actions by estimating the object's 6-Degrees of Freedom (DoF) — its position and orientation — from RGB-D (color and depth) camera input. While traditionally treated as a standalone module, our framework integrates it directly as a ROS2 SW component orchestrated by the AI agent, which enables seamless coordination with perception, planning, and task sequencing.

While not fully autonomous, the proposed framework enables the interpretation of structured or constrained NL instructions and maps them to modular ROS2 SW components for perception, grasp generation, motion planning, and task sequencing. The embodied AI agent leverages open-source LLMs and VLMs to parse commands, interpret visual context, and orchestrate tool invocation across the ROS-based stack. Evaluated through real-world assembly tasks, such as vision-guided cable manipulation, visual feedback, and task reconfiguration, the framework demonstrates how integrating semantic reasoning with modular control supports the vision of agentic AI for flexible and scalable assembly automation in manufacturing [21].

3. Embodied AI Agent Framework

The framework introduces an embodied AI agent as a task-level *orchestrator* that intelligently sequences existing ROS2 SW tools—such as MoveIt2, the MTC, and Contact-GraspNet [27] based on user input. These widely adopted open-source components offer robust capabilities for motion planning, grasping, and perception. By integrating them under a unified architecture powered by LLMs and VLMs, the agent enables semantic-level control over robotic execution, replacing traditional state machine reprogramming with NL language or voice-based interaction.

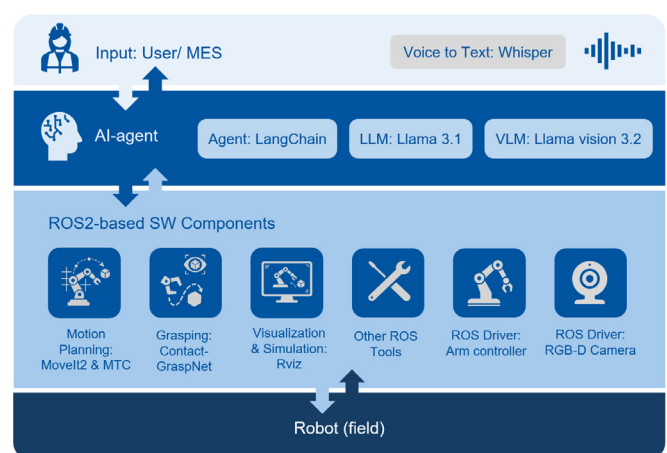


Fig. 1: Layered architecture of the embodied AI agent framework.

Table 1: Comparison of related frameworks with the proposed approach.

Criteria	ROS-LLM [23]	ROSA [24]	PickGPT [25]	RAI [26]	Ours
Planning Strategy	Scripted Sequences	Manual Mapping	Closed-loop	Prompt-based + Tools	AI-driven Orchestration
Task Decomposition	Manual	Partial	Limited	LLM-only	LLM + Agent
Modular ROS 2 Integration	Partial	Static Actions	No	Connector-based	Modular, ROS2-native
Use Case Domain	Domestic	General	Logistics	Multi-agent	Assembly Automation
Vision Integration	No	No	Yes	Yes (via tools)	Yes (VLM-enabled)
Task Sequencing (MTC)	No	No	No	Not supported	Supported
Non-Expert Usability	Yes (limited)	Yes (with setup)	Yes (proprietary)	Partial (requires setup)	Yes (voice & text)
LLM/VLM Type	Cloud-based	Cloud/Open-source	Proprietary	Cloud-based	Open-source (e.g., Llama)

3.1. Framework Overview

Figure 1 presents the layered architecture of the framework. At the top layer, a human operator or even a Manufacturing Execution System (MES) issues instructions via speech or text. A central agent, implemented using LangChain [28] and powered by Meta’s Llama 3.1 and Llama Vision 3.2 [29], processes spoken or written instructions and maps them to tool invocations from ROS2 SW components.

The core of the framework is a modular action agent capable of calling both atomic and compound behaviors, such as pose transitions, gripper control, vision capture, grasp generation, and motion execution. The agent then invokes the low-level ROS2 SW components like perception, planning, visualization, and actuation. This includes drivers for the robot arm and its integrated RGB-D camera, as well as key libraries such as MoveIt2, the MTC, and Contact-GraspNet. The SW stack is modularized using Docker containers, ensuring reproducibility and modular deployment across simulation, testing (Rviz), and real-world scenarios.

3.2. Open-Source LLM and VLM Integration

Open-source LLMs and VLMs present a cost-effective and customizable alternative to commercial Application Programming Interface (API)s. However, their deployment requires sufficient GPU resources.

Llama 3.1 (8B parameters) is chosen as the primary LLM due to its compact size and robust capabilities for NL parsing and tool invocation. It enables the AI agent to interpret constrained voice or text instructions and match them to a predefined set of ROS2-based SW tools (components). To improve performance, prompt engineering techniques are used to guide model behavior and ensure reliability under varying inputs. Prompts are also designed to prevent unsafe actions and handle ambiguities conservatively.

For multimodal inputs, **Llama Vision 3.2** (11B parameters) is employed as the VLM to interpret RGB-D images captured from the robot’s wrist inbuilt camera. The model supports scene captioning, object listing, and visual reasoning. Similar to the LLM, its outputs are shaped using prompt engineering to enforce consistency and restrict hallucinated or irrelevant descriptions. This allows users to query the robot with instructions such

as “describe what you see” and receive interpretable feedback grounded in the real environment.

3.3. End-to-End Workflow

Figure 2 illustrates two representative interactions handled by the framework. In the first, the user issues a compound instruction such as “go to home pose, then open the gripper.” Spoken commands are transcribed via Whisper [30] and parsed by the LLM to identify the required tools, which the agent then invokes via ROS2 SW component, such as MoveIt2 for trajectory planning and the arm controller for execution.

In the second case, the user issues a perception request like “describe what you see.” The agent captures a scene image using the robot’s vision module and forwards it to the VLM. The VLM returns a structured textual description of the environment, listing recognizable objects and spatial relations. This allows the robot to provide semantic feedback or use that information for downstream planning (e.g., picking the object in front).

3.4. Tool-Oriented Agent Features

The developed framework features a set of over 20 ROS2-integrated tools as callable functions, each defined with structured input-output interfaces. These tools span low-level primitives (e.g., pose movement, gripper actuation, base rotation), compound behaviors (e.g., pick-and-place routines via Contact-GraspNet and MTC), perception utilities (e.g., image capture, VLM-based interpretation), and system functions (e.g., trajectory replay, emergency stop).

To enable flexible orchestration, tool invocation follows a reasoning loop: (1) *Input* → (2) *Parse* → (3) *Resolve tools* → (4) *Call tool* → (5) *Monitor execution* → (6) *Provide feedback*. This abstraction allows chaining multiple tools into semantically meaningful behaviors without requiring hardcoded state machines or domain-specific scripting.

3.5. Grasping Component: Integrating Contact-GraspNet

To support robust pick-and-place operations, the framework integrates Contact-GraspNet [27], a deep learning-based grasp detection model that infers a fixed number of 6-DoF grasp poses with confidence scores from RGB-D input. Unlike traditional

sion stack and Contact-GraspNet for grasp generation, followed by execution via the MoveIt2 & MTC.

(3) *Visual scene feedback*, where the user asks the robot to “describe what you see”, prompting the robot to capture an RGB-D image, invoke the VLM (Llama Vision 3.2) for interpretation, and return a NL description of the scene.

(4) *Task-cycle recording for scalable execution*, where the agent records multiple user-commanded trajectories in the background and replays them on demand. This feature allows operators to first validate motions interactively and then repeat the exact task-cycle numerous times without further manual input, reflecting a typical requirement in production.

5. Results and Discussion

This section presents the experimental results for each of the defined scenarios. In this context, a *trial* refers to one full execution of the desired task sequence under test conditions.

5.1. Scenario (1) - Basic pose control via voice & text

Across 10 trials, all commands were successfully parsed by the LLM and executed by the corresponding ROS2 primitives, achieving 100% interpretation and execution accuracy. The agent responded within 1 second in all cases, with total task completion times ranging from 0.7 to 22 seconds depending on the motion. The average completion time across all commands was 8.7 s, as summarized in Table 2. The results indicate that the framework achieved responsive operation and consistent performance during low-level teleoperation tasks via voice-to-action. Notably, all voice commands could equivalently be issued through text (e.g., via chat interface), offering flexibility in interaction. Moreover, the LLM was able to correctly interpret and match user intent even in the presence of minor speech errors or grammatical inconsistencies, highlighting the flexibility of the agent’s NL interface.

Table 2: Scenario (1): Basic pose control via voice and text (10 trials). Exec. = agent response latency; Total = full motion duration.

Command	GPU PC (RTX 4090)		Laptop (GTX 1660)	
	Exec. (s)	Total (s)	Exec. (s)	Total (s)
Move to Zero Pose	1.0	22.0	9.0	22.0
Move to Home Pose	1.0	22.0	9.0	22.0
Move to Retract Pose	1.0	6.0	9.0	6.0
Move Downward	1.0	5.0	9.0	5.0
Close Gripper	1.0	0.7	9.0	0.7
Move Left	1.0	4.0	9.0	4.0
Move Upward	1.0	4.2	9.0	4.2
Move Right	1.0	4.0	9.0	4.0
Back to Home Pose	1.0	6.0	9.0	6.0
Stop	1.0	0.0	6.0	0.0
Average	1.0	8.7	8.7	8.7

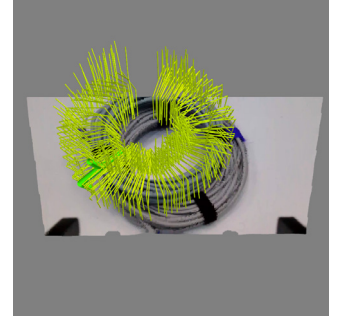
5.2. Scenario (2) - Cable grasping

Across 30 trials, the agent successfully detected and picked two cables (per trial) in 21 cases (70%), using the developed

grasp generation pipeline and MTC for trajectory planning. Failures in 9 trials were primarily due to either MTC failing to compute collision-free trajectories or the robot entering safety mode due to self-collisions during execution. In 6 of these cases, valid grasps were generated for one or both cables, but trajectory planning failed. Notably, when only one cable was picked (4 trials), the second grasp often failed due to occlusions or unreachable poses.



(a) Robot arm in home pose.



(b) Generated grasps poses.



(c) Robot arm approaching a cable.



(d) Robot arm picking up the cable.



(e) Robot arm moving to the target.



(f) Robot arm placing the cable.

Fig. 4: Scenario (2): Cable grasping.

Table 3: Scenario (2): Cable grasping across 30 trials (2 cables per trial).

Metric	Result
Total Trials	30
Cables Targeted (2 per trial)	60
Cables Successfully Picked	42 (70%)
Primary Failure Causes	7: Task Constructor failed 5: Safety triggers (self-collisions)
Average Grasp Generation Time	8.4 s
Average Execution Time per Cable	19.2 s
Partial Success (1 cable only)	4 trials
Complete Failures (0 cables)	5 trials

The grasp generation time was measured from the moment an image was saved until the generated grasps were visualized on the screen. On average grasp generation took 8.4 s while execution time per cable was 19.2 s. A summary of success rates, failure modes, and timing is presented in Table 3, while Figure 4 shows the grasping pipeline execution on the real robot.

5.3. Scenario (3) - Visual Scene Feedback

Across 20 trials, the framework returned coherent and contextually accurate descriptions of objects and their spatial relations with a 100% success rate. An example of this process is shown in Figure 5, where the robot perceives a coiled cable and generates a NL description. However, the LLM–tool integration was not always consistent in triggering the visual feedback pipeline for the query “describe what you see,” occasionally requiring few attempts before the associated functions executed. The average inference time on an RTX 4090 device was 1.67 s, enabling near real-time visual feedback. In contrast, on a GTX 1660 laptop, processing times increased drastically, averaging 336 s (5.6 minutes) per query. This performance disparity, summarized in Table 4, highlights the importance of adequate GPU resources for local VLM deployment. The visual scene feedback capability demonstrates the agent’s potential for enabling high-level situational awareness and grounding NL scenes in the environment, which can support downstream applications such as object-specific planning or failure explanation.

```
[INFO] [1753881960.092908391] [Kinova_Agent]: Waiting for the next message...
[INFO] [1753881961.023119760] [Kinova_Agent]: Waiting for message to be published ...
[INFO] [1753881961.023352835] [Kinova_Agent]: Received message: describe what you see
[INFO] [1753881961.023592891] [Kinova_Agent]: Invoking Agent with the query: [describe what you see]
[INFO] [1753881961.366106345] [image_capture_node]: Image successfully saved as: /home/ubuntu/worksp
aces/kinova_ws/src/kinova-ros2/src/kinova_agent/images/image.png
[INFO] [1753881961.366274249] [image_capture_node]: Processing the saved image...
[INFO] [1753881966.914202554] [image_capture_node]: The image shows a coiled electrical cable, wrappe
d tightly with black tape and secured with red tapes at each end. The cable is white and appears to
be made of this wire. It is neatly wound into a circle or coil shape, held together by the black t
ape, which also secures it in place.

In the background, there is a plain white surface that could be a table or other flat surface. The o
verall atmosphere suggests a professional setting, possibly an office or workshop where electrical e
quipment is being used or maintained.
```

Fig. 5: Scenario (3): Visual scene feedback example. The robot perceives the real-world object (cable) and generates a NL description with integrated VLM.

Table 4: Scenario (3): Visual scene feedback (20 trials). Avg. time is for generating descriptions from RGB-D images.

Metric	GPU PC (RTX 4090)	Laptop (GTX 1660)
Trials Executed	20	3
Success Rate (%)	100	100
Avg. Inference Time (s)	1.67	336 (5.6 min)
Min/Max Time (s)	1.5 / 2.1	310 / 360

5.4. Scenario (4) - Task-cycle recording for scalable execution

In this scenario, the AI-agent received compound NL commands, such as “Move to retract pose and then move left” or “Rotate gripper clockwise and move forward”. These commands were executed in sequence while the corresponding joint trajectories were recorded. A total of 15 trials were performed, covering a diverse set of motion combinations including linear translations, pose changes, and limited end-effector rotations.

The task replaying was successful in 13 out of 15 cases, with all intended motions accurately reproduced. In two instances, only partial execution occurred due to safety triggers, typically when the arm transitioned quickly between complex poses. Performance, including success rates, number of commands, and common failure causes, are summarized in Table 5.

While the replayed motion closely matched the original trajectories, a notable limitation emerged: gripper actions (i.e., open/close) were not captured or replayed during the recording phase. As a result, this mechanism could not be applied to validate the grasping pipeline from Scenario (2), which requires accurate end-effector control for successful object manipulation.

Table 5: Scenario (4): Task-cycle recording and replay (15 trials).

Metric	Result
Total Trials	15
Commands Recorded	37
Full Replays (success)	13 (87%)
Partial Replays (safety triggers)	2 (13%)
Avg. Commands per Trial	2.5
Gripper Actions Recorded	No
Failure Cause (most common)	Safety triggers during pose transitions

Across all four scenarios, the agent demonstrated reliable task execution, achieving consistently high success rates in basic commands and visual scene interpretation. The grasping tasks revealed occasional failure modes, primarily related to trajectory planning and self-collisions. Replay functionality worked reliably in all tested sequences. Importantly, inference latency for vision models was strongly hardware-dependent, with VLM responses averaging under 2 seconds on RTX 4090 but exceeding 5 minutes on a GTX 1660. These results suggest the potential feasibility of embodied AI agents for flexible assembly tasks and highlight deployment considerations for real-time performance.

6. Conclusion and Outlook

This work presented a modular embodied AI agent framework integrating open-source LLMs, VLMs, and ROS2 SW components for NL-based robotic task execution. By orchestrating perception, grasping, and motion tools, the framework enables reprogrammable assembly workflows accessible to non-experts.

Evaluated across four scenarios, the AI-agent showed high success rates and low latency in motion control, scene interpretation, and semantic grasping. Trajectory recording and replay supported task reuse, though gripper actions were not captured. Scene feedback was accurate and fast on GPU hardware, but limited devices revealed deployment challenges.

Key future directions include: (1) investigating voice vs. text input limitations in noisy or ambiguous settings, as robust interaction is critical for shop-floor usability; (2) exploring LLM/VLM swapping strategies to enable multilingual support and flexible upgrades, allowing models to be replaced with faster or more capable alternatives for time-critical tasks and to integrate improved multimodal models as they emerge; and (3)

enhancing grasping through active perception and confidence-driven planning to boost reliability in cluttered, dynamic environments.

While the framework already decomposes NL instructions into ROS2 actions and uses LLM/VLM models for grounding, further work is needed to make this process more adaptive for complex, multi-step tasks. Future research should also benchmark which models perform best for assembly-specific grounding. Adding semantic memory, allowing the agent to retain task and object context, could improve continuity across extended tasks. Extending the framework to mobile platforms would further increase flexibility by enabling operation in larger and more dynamic workspaces. The recording and replay feature also points toward voice-guided programming. Finally, cloud-based or hybrid edge–cloud execution, supported by low-latency networks (e.g., 5G), may provide scalable solutions for demanding reasoning and orchestration tasks. Overall, results demonstrate the embodied AI-agent as a promising approach for intuitive, reprogrammable robotics in dynamic assembly, advancing flexible and accessible automation.

Acknowledgements

Parts of this research were funded by the EU project TARGET-X. The TARGET-X project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation program under Grant Agreement No: 101096614. The authors are responsible for the content of this publication.

References

- [1] F. Bärmühl, S. F. Schäfer, A. Hohmann, R. Daub, Short-cycled factory planning – motivation and existing challenges, *Procedia CIRP* 130 (2024) 1708–1713, 57th CIRP Conference on Manufacturing Systems 2024 (CMS 2024).
- [2] A3 Association for Advancing Automation, [Upskilling and automation: Two proven solutions to the labor shortage](https://www.automate.org/robotics/industry-insights/upskilling-automation-labor-shortage-solutions) (2025). URL <https://www.automate.org/robotics/industry-insights/upskilling-automation-labor-shortage-solutions>
- [3] B. Starly, A. Angrish, P. Cohen, Research directions in democratizing innovation through design automation, one-click manufacturing services and intelligent machines (2019). [arXiv:1909.10476](https://arxiv.org/abs/1909.10476).
- [4] B. Nitsche, Unravelling the complexity of supply chain volatility management, *Logistics* 2 (3) (2018).
- [5] G. Hüttemann, Modellbasierte a priori bewertung von linienlosen mobilen montagesystemen: Lehrstuhl für fertigungsmesstechnik und qualitätsmanagement / werkzeugmaschinenlabor wzl der rwth aachen, Ph.D. thesis, Apprimus Verlag and Dissertation, RWTH Aachen University, 2020.
- [6] Y. Zang, M. Qu, D. T. Pham, R. Dixon, F. Goli, Y. Zhang, Y. Wang, Robotic disassembly of electric vehicle batteries: Technologies and opportunities, *Computers Industrial Engineering* 198 (2024) 110727.
- [7] D. Gebauer, J. Dirr, G. Reinhart, Concept for robot-based cable assembly regarding industrial production, in: T. Schüppstuhl, K. Tracht, A. Raatz (Eds.), *Annals of Scientific Society for Assembly, Handling and Industrial Robotics 2021*, Springer International Publishing, Cham, 2022, pp. 113–124.
- [8] Z. Jin, R. M. Marian, J. S. Chahl, Achieving batch-size-of-one production model in robot flexible assembly cells, *The International Journal of Advanced Manufacturing Technology* 126 (5) (2023) 2097–2116.
- [9] R. Ghzouli, T. Berger, E. B. Johnsen, A. Wasowski, S. Dragule, Behavior trees and state machines in robotics applications (2023). [arXiv:2208.04211](https://arxiv.org/abs/2208.04211).
- [10] M. Iovino, J. Förster, P. Falco, J. J. Chung, R. Siegwart, C. Smith, Comparison between behavior trees and finite state machines (2024). [arXiv:2405.16137](https://arxiv.org/abs/2405.16137).
- [11] A. Sidorenko, M. Rezapour, A. Wagner, M. Ruskowski, Towards using behavior trees in industrial automation controllers (2024). [arXiv:2404.14030](https://arxiv.org/abs/2404.14030).
- [12] T. Yu, Q. Chang, Reinforcement learning based user-guided motion planning for human-robot collaboration (2022). [arXiv:2207.00492](https://arxiv.org/abs/2207.00492).
- [13] J. Cloete, W. Merkt, I. Havoutis, Adaptive manipulation using behavior trees (2025). [arXiv:2406.14634](https://arxiv.org/abs/2406.14634).
- [14] F. Mo, F. M. Monetti, A. Torayev, et al., A maturity model for the autonomy of manufacturing systems, *The International Journal of Advanced Manufacturing Technology* 126 (2023) 405–428.
- [15] M. Albus, M. F. Huber, Resource reconfiguration and optimization in brownfield constrained robotic assembly line balancing problems, *Journal of Manufacturing Systems* 67 (2023) 132–142.
- [16] M. Soori, B. Arezoo, R. Dastres, Artificial intelligence, machine learning and deep learning in advanced robotics, a review, *Cognitive Robotics* 3 (2023) 54–70.
- [17] J. T. Licardo, M. Domjan, T. Orehovalčki, Intelligent robotics—a systematic review of emerging technologies and trends, *Electronics* 13 (3) (2024).
- [18] M. Soori, F. K. G. Jough, R. Dastres, B. Arezoo, Ai-based decision support systems in industry 4.0, a review, *Journal of Economy and Technology* (2024).
- [19] J. Fan, Y. Yin, T. Wang, W. Dong, P. Zheng, L. Wang, Vision-language model-based human-robot collaboration for smart manufacturing: A state-of-the-art survey, *Frontiers of Engineering Management* 12 (1) (2025) 177.
- [20] H. Jeong, H. Lee, C. Kim, S. Shin, A survey of robot intelligence with large language models, *Applied Sciences* 14 (19) (2024).
- [21] Y. Ren, Y. Liu, T. Ji, X. Xu, Ai agents and agentic ai-navigating a plethora of concepts for future manufacturing (2025). [arXiv:2507.01376](https://arxiv.org/abs/2507.01376).
- [22] W. Huang, P. Abbeel, D. Pathak, I. Mordatch, Language models as zero-shot planners: Extracting actionable knowledge for embodied agents (2022). [arXiv:2201.07207](https://arxiv.org/abs/2201.07207).
- [23] 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, Y. Zhuang, K. Shao, X. Quan, J. Hao, J. Wang, H. Bou-Ammar, Ros-llm: A ros framework for embodied ai with task feedback and structured reasoning (2024). [arXiv:2406.19741](https://arxiv.org/abs/2406.19741).
- [24] R. Royce, M. Kaufmann, J. Beckett, S. Moon, K. Carpenter, K. Pak, A. Towler, R. Thakker, S. Khattak, Enabling novel mission operations and interactions with rosa: The robot operating system agent (2025). [arXiv:2410.06472](https://arxiv.org/abs/2410.06472).
- [25] Sereact GmbH, [Pickgpt – a robotics transformer that changes the world of logistics](https://sereact.ai/posts/pickgpt-a-large-language-model-for-generalized-robot-manipulation), accessed: 2025-07-07 (2023). URL <https://sereact.ai/posts/pickgpt-a-large-language-model-for-generalized-robot-manipulation>
- [26] K. Rachwał, M. Majek, B. Boczek, K. Dabrowski, P. Liberadzki, A. Dabrowski, M. Ganzha, Rai: Flexible agent framework for embodied ai (2025). [arXiv:2505.07532](https://arxiv.org/abs/2505.07532).
- [27] M. Sundermeyer, A. Mousavian, R. Triebel, D. Fox, Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes (2021). [arXiv:2103.14127](https://arxiv.org/abs/2103.14127).
- [28] L. AI, [Langchain: Build context-aware reasoning applications](https://github.com/langchain-ai/langchain), accessed: 2025-07-27 (2025). URL <https://github.com/langchain-ai/langchain>
- [29] Ollama, [Ollama: Get up and running with llama 3.3, deepseek-r1, phi-4, gemma 3, mistral small 3.1 and other large language models](https://ollama.com/), accessed: 2025-07-27 (2025). URL <https://github.com/ollama/ollama>
- [30] OpenAI, Whisper: General-purpose speech recognition model, <https://github.com/openai/whisper>, accessed: 2025-07-24 (2022).