Agentic AI for Robot Control: Flexible but Still Fragile

Oscar Lima[1,2] Marc Vinci[1,2] Martin Günther[1] Marian Renz[1,2]
Alexander Sung[1] Sebastian Stock[1] Johannes Brust[1] Lennart Niecksch[1,2]
Zongyao Yi[1,2] Felix Igelbrink[1,2] Benjamin Kisliuk[1,2]
Martin Atzmueller[2,1] Joachim Hertzberg[2,1]

[1]German Research Center for Artificial Intelligence (DFKI), Osnabrück, Germany
[2]Osnabrück University, Semantic Information Systems Group, Osnabrück, Germany

AAAI Spring Symposium on Machine Learning and Knowledge Engineering

Motivation & Problem Statement

- **The Promise:** Modern LLMs possess vast commonsense knowledge and reasoning capabilities, making them highly attractive for open-vocabulary robotic planning.
- **The Challenge:** Bridging abstract, language-based plans to real-time physical robot control under uncertainty, partial observability, and sensor noise.
- **Our Approach:** An *agentic architecture* that leverages LLMs in a closed plan-act-monitor loop.
- **Core Finding:** The system is highly **flexible** (easily transferable between completely different robot domains) but remains **fragile** in execution (highly sensitive to prompts and non-deterministic).

Agentic AI vs. Vision-Language-Action (VLA) Models

Addressing recent trends in Embodied AI...

End-to-End VLA Models
(e.g., OpenVLA, RT-2)

- Map raw perception directly to low-level control policies.
- Highly capable at learned manipulation tasks.
- Opaque latent spaces. Hard to formally verify or impose hard safety constraints.

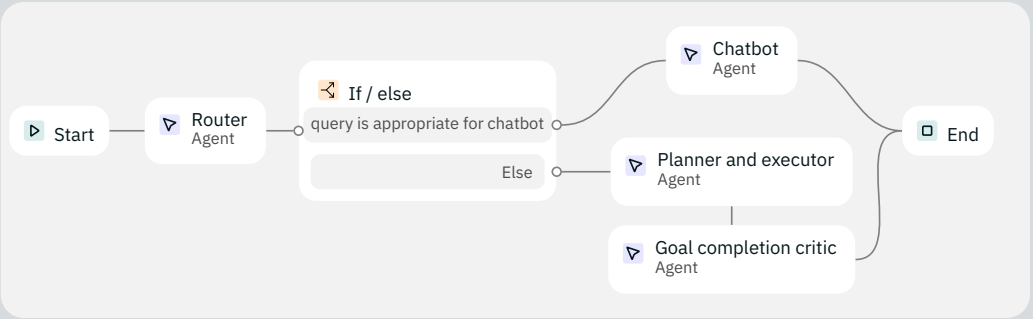
Our Agentic Architecture
(Decoupled Reasoning)

- LLM operates over an action API and verified symbolic state predicates.
- Relies on existing robust robot stacks (ROS, MoveIt).
- Transparent. Exposes structured introspection of the decision process.

Multi-Agent Control Architecture

We decompose the problem using specialized agents (implemented via OpenAI tools):

- **Router Agent:** Conversational vs. actionable.
- **Chatbot Agent:** Basic queries (no actions).
- **Planner/Executor:** Plan-act-monitor loop.
- **Critic:** Safety check for hallucinated completion.



The Planner and Executor: Tools & Interface

The LLM does not output raw joint velocities; it uses a strictly constrained API:

- `reflect` Emits structured introspection for debugging, without altering the environment (serves as planning space).
- `act` Executes exactly one high-level robot action (e.g., `navigate`, `pick`, `place`, `speak`).
- `get_semantic_snapshot` Returns the current sensor-driven symbolic state.
- `check_events` Queries for discrete, text-based exogenous events (e.g., battery warnings, emergency stops).

State Abstraction & Symbolic Fact Generation

How does the LLM know the state of the world?

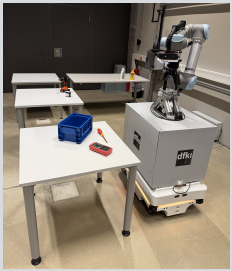
We utilize a lightweight **Symbolic Fact Generation** pipeline to map raw sensors to discrete predicates:

- **Indoor Manipulator:** Object detection & pose estimation translate to predicates like `on(multimeter_1, table_3)` or `arm_posture(observe)`.
- **Outdoor Robot:** GPS & telemetry translate to logical zones (`at(poi_1)`) and battery states (`okay`, `low`).

Why? It enables uniform reasoning across vastly different hardware using natural-language-like assertions, reducing hallucinations.

Heterogeneous Robotic Platforms

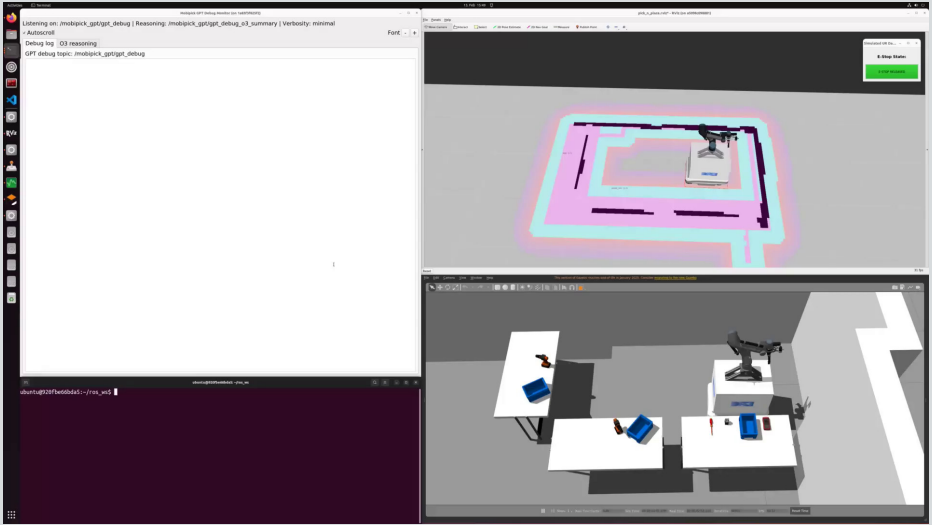
Evaluated in both simulation (Gazebo) and real-world deployment:



Mobipick (Indoor)
Mobile manipulator (6-DoF arm).
Tasks: Search, pick, container insertion, placement.

Valdemar (Outdoor)
Agricultural monitoring robot.
Tasks: Long-range navigation, crop scanning, battery management.

Video Demonstration



Findings: The System is Highly Flexible

- **Zero-Shot Domain Transfer:** Moving from the indoor Mobipick to the outdoor Valdemar required *almost exclusively prompt updates* (domain model, action catalog) and rebinding the same tool schema to a different API.
- **Ambiguity Management:** Given the command *“Pick any tool I can use to screw with”*, it successfully identifies and selects a screwdriver over a drill based on commonsense reasoning.
- **Refuting Invalid Commands:** When told *“Fly to Paris”*, the agent logically rejects it: *“I am a ground-based farm robot. I can’t fly, and Paris is outside my operation area.”*

Findings: The System is Still Fragile

Despite successful execution, long-horizon autonomy remains brittle:

- **Prompt Sensitivity:** Operational constraints often must be stated redundantly across the prompt (instructions, heuristics, memory) to guarantee adherence.
- **Event Polling Limitations:** Asynchrony is hard. The LLM must explicitly poll `check_events`. Hard preemption of long-running navigation tasks remains a systems engineering bottleneck.
- **“Verbal-only” Failures:** Occasional alignment failures where the agent *explains* how to solve the task perfectly, but issues zero `act` tool calls. (Necessitating the Goal Completion Critic).

Addressing Limitations & Future Work

Based on peer review and ongoing research, our next steps entail:

- **Quantitative Benchmarking:** Moving from qualitative proof-of-concepts to large-scale sim-to-real evaluations to isolate physical perception failures from LLM reasoning failures.
- **Prompt Ablation Studies:** Systematically removing affordances, heuristics, or failure memory to measure their direct impact on success rates.
- **Scalability:** Dynamically generating tool schemas from robot capability descriptions rather than using hand-wired action catalogs.
- **On-Device LLMs:** Mitigating latency and network dependency by running smaller reasoning models locally.

Conclusion

- By interleaving LLM deliberation with concrete robot skills, we can achieve **transparent, interpretable, and adaptable** robotic control.
- The agentic approach is highly modular, easily adapting to vastly different embodiments (indoor arms vs. outdoor tractors).
- However, **robustness is not yet solved**. Solving prompt sensitivity and asynchronous event handling remains key to moving these systems out of the lab and into the real world.

Thank You! Questions?

Code, Prompts, and Video available at:
`https://dfki-ni.github.io/AGENTS-MAKE-2026`