Demo: Swarm Chess — Grounding LLM Agents in a Physical Robot Swarm

Kunfu Wang*
kunfuwang@hkust-gz.edu.cn
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Peiji Li*
lipeiji550@gmail.com
Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China
Shenzhen, China

Yinghao Gao
ygao651@connect.hkust-gz.edu.cn
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Jiayang Liu
jliu045@connect.hkust-gz.edu.cn
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Jiezhen Dai
jdai942@connect.hkust-gz.edu.cn
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Tengfei Chang†
tengfeichang@hkust-gz.edu.cn
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Abstract

We present Swarm Chess, a physical robot-swarm demonstration platform built using a fleet of AtomBot mobile robots. The system integrates lighthouse localization [1], BLE broadcast communication, and an application-layer API that packages robot perception and control as callable tools for LLM agents' online reasoning. Our Web UI coordinates the wall and floor displays, visualizing the physical board, agent reasoning, tool calls, and robot motion.

In this demo, we show two interaction modes: Agent vs. Agent, where two LLM agents autonomously play chess through the robot swarm, and Agent vs. Human, where visitors can issue moves and observe how symbolic decisions are executed by physical robots.

CCS Concepts

• **Computer systems organization** → **Robotics**; • **Computing methodologies** → *Multi-agent systems.*

Keywords

swarm robotics, LLM agents, lighthouse localization, BLE

1 Introduction

LLM agents provide a new interface for embodied systems by combining reasoning and acting in an interleaved loop [4]. In such a loop, observations from the environment can be incorporated into the agent's context, while actions are issued through external tools or API calls [2]. This capability is especially appealing for interactive demonstrations, where visitors can observe how symbolic reasoning is translated into physical robot motion. A key challenge for such demonstrations is connecting both autonomous agent decisions and human natural language commands to a live physical state that changes as robots move.

Swarm Chess explores this loop in a familiar tabletop game. Each chess piece is represented by a mobile robot, turning symbolic chess decisions into physical multi-robot motion. The demo uses this setting to show how LLM agents can reason over a structured task state while acting through a robot swarm.

*Kunfu Wang and Peiji Li contributed equally to this work.
†Corresponding author.

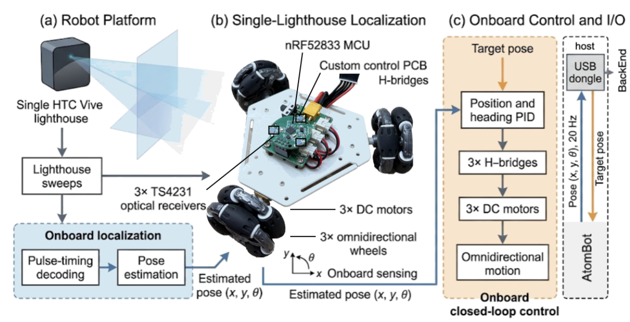

Figure 1: AtomBot mobile robot platform, localization, and onboard control.

2 System Overview

2.1 Robot Platform and Localization

AtomBot is a compact omnidirectional mobile robot built around an nRF52833 microcontroller running bare-metal firmware. AtomBot performs onboard sensing, wireless communication, and closed-loop position control directly on the MCU. The custom control PCB drives three DC motors through H-bridges, each coupled to an omnidirectional wheel, while onboard PID loops regulate position and heading.

For localization, each robot carries three TS4231 optical receivers and computes its own pose from HTC Vive lighthouse sweeps [1]. The firmware decodes pulse timing and estimates the robot pose before broadcasting the resulting position and heading. Pose packets are sent as BLE broadcasts at 20 Hz, received by a USB dongle, and forwarded over a serial interface to the backend. The backend exposes robot state through an SSE stream and accepts target commands through HTTP APIs, hiding the lower-level wireless and serial transport from the chess agent.

2.2 Grounded Chess Agent

The agent core follows a ReAct loop [4] using the Xiaomi MiMo-V2.5-Pro API. At each step, the LLM reasons over the task context, decides whether to call an external tool, incorporates the returned

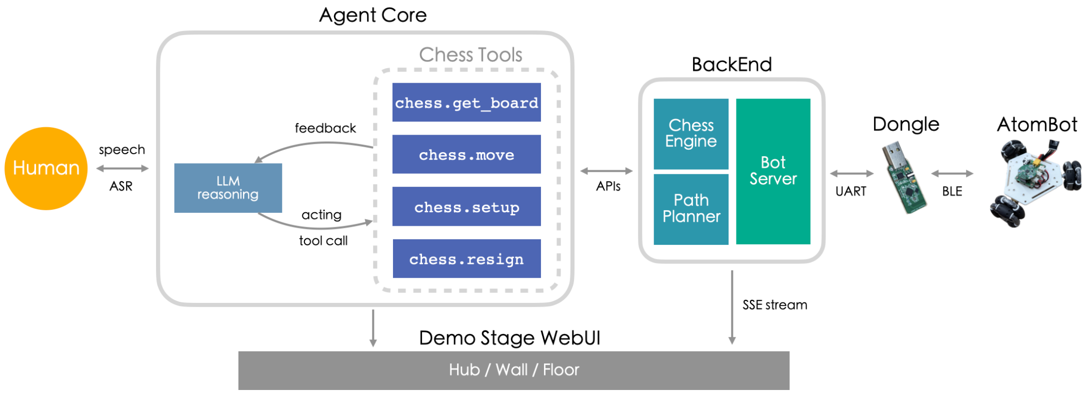

Figure 2: System overview of Swarm Chess.

tool result, and continues until it produces a move or a final response. This tool-use mechanism [2] allows the chess agent to use live system state during online decision making rather than relying only on its prior context.

In Swarm Chess, the agent does not access robot identifiers, metric coordinates, or serial commands directly. Instead, the robot backend is wrapped as four chess tools:

- `chess.get_board`: reads the current physical board.
- `chess.move`: requests a legal chess move.
- `chess.setup`: arranges or resets the pieces.
- `chess.resign`: ends an agent game.

All physical chess actions are submitted to the chessboard management engine, which validates each move and converts it into goal positions. The resulting multi-agent path finding (MAPF) problem is solved using WHCA* [3]. The planner sends target points to the robots for point-to-point execution.

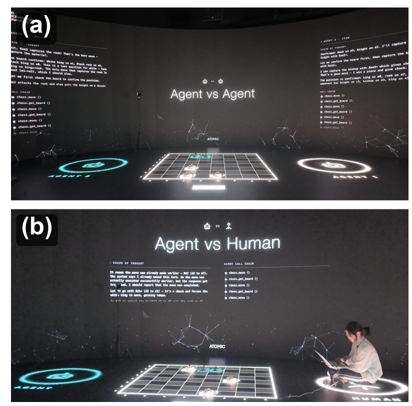

Figure 3: Swarm Chess demo stage in (a) Agent vs. Agent and (b) Agent vs. Human modes.

3 Demonstration

The demo presents Swarm Chess as a live robot swarm chess game on a chessboard displayed on the floor screen. The Demo Stage Web UI is organized into three synchronized views: a control hub, a wall display, and a floor display. The control hub manages the game mode, board state, and operator actions. The wall display shows each agent's chain of thought and tool call history, while the floor display shows the chessboard and robot execution. The visual design uses a luminous sci fi style: the board changes color when the turn switches between players, and a light ring is rendered beneath each robot to indicate its current chess role.

The setup requires a wall display and a floor display, implemented either with projection or physical screens. Because the demo stage is built as a Web UI, the host computer, display clients, and backend services require a reliable local area network. Our calibration tool chain maps the screen coordinates to the physical play area, allowing the displayed chessboard geometry to match the measured robot distances.

We demonstrate two interaction modes. In Agent vs. Agent mode (Fig. 3(a)), two LLM agents alternately observe the board, reason about the next move, and execute moves through the robot swarm. The chess engine maintains the game state, checks legal moves, and automatically switches turns between the agents. In Agent vs. Human mode (Fig. 3(b)), a visitor plays against the agent using speech input. An operator agent interprets the human natural language command and issues the corresponding chess tool call on the visitor's behalf. In both modes, the agents treat the AtomBot robots as physical chess pieces. Accepted moves are grounded in the measured board state. For a capture, the capturing robot first approaches the occupied destination. Once it is close, the engine records the capture and sends the captured robot to an off-board graveyard. The capturing robot then enters the cleared square.

References

[1] Said Alvarado-Marin, Cristobal Huidobro-Marin, Martina Balbi, Trifun Savić, Thomas Watteyne, and Filip Maksimovic. 2024. Lighthouse Localization of Miniature Wireless Robots. *IEEE Robotics and Automation Letters* (2024). doi:10.1109/LRA.2024.3405345

[2] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[3] David Silver. 2005. Cooperative Pathfinding. In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. AAAI Press, 117–122. doi:10.1609/aiide.v1i1.18726

[4] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*.