From Speech to Delivery: Human-Centered Multimodal Interaction for Indoor Service Robotics

Annika An[*] Zeyi Chen[*] Harshavardhan Reddy Gajarla[*]
Xinyi Hu[*] Rishabh Kumar[*] Ningbo Li[*]
Yifan Li[*] Ray Lin[*] Anshul Prakash[*]
Linzhengrong Shao[*] Siyang Shen[*] Matthew Taruno[*]
Chenghao Wang[*] Hanyang Wang[*] Fiona Wu[*]
Zhenglong Yang[*] Yuxin Zhang[*] Yuzhe Zhang[*]
Jack Hatcher Zubin A Assadian Haonan Peng
John Raiti

[*]All authors contributed equally to this research.
Global Innovation Exchange, University of Washington
Seattle, WA, USA

Abstract

This paper presents the design and implementation of a voice-controlled autonomous robotic delivery system that integrates physical human-robot interaction (HRI), perception, manipulation, and navigation. At the core of the system is an orchestration module implemented as a finite state machine, responsible for managing task execution and coordinating subsystem communication through ROS2 topics and services. The robot interprets natural language commands using a GPT-based Physical AI module, detects objects via perception input, manipulates them with a robotic arm, and navigates to user-defined destinations. We describe the full task pipeline—from voice input to final delivery—highlighting the orchestration logic, system robustness strategies, and real-time feedback mechanisms. Our results demonstrate that modular ROS2-based orchestration enables reliable multi-step execution in collaborative HRI scenarios.

CCS Concepts

• **Computer systems organization** → **Robotics**; • **Human-centered computing** → **Human computer interaction (HCI)**; *Interaction paradigms*; • **Computing methodologies** → *Natural language processing*; *Intelligent agents.*

Keywords

Human-Robot Interaction, Physical AI, Mobile Manipulation, Multi-Modal Integration

HRI, Edinburgh, Scotland

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

As robotics advances toward real-world autonomy, a central challenge in Human-Robot Interaction (HRI) is enabling robots to interpret and execute tasks from intuitive, human-centered input. Although perception, manipulation, and navigation have individually seen significant progress, integrating them into a unified system that responds fluidly to human intent remains difficult. We address this challenge by developing a voice-controlled robotic delivery system that allows users to issue natural-language commands (e.g., "Bring me an apple"), which the robot interprets through a Physical AI interface before autonomously detecting, manipulating, transporting, and delivering the object with real-time feedback. Achieving this requires tight coordination among ROS2 subsystems for perception, manipulation, and navigation, guided by a centralized orchestration module that sequences actions, monitors readiness, and handles asynchronous feedback and partial failures.

Our system builds on prior work in visual docking and servoing, where Saputra et al. use non-linear model predictive control with ArUco feedback [6] and Hong et al. enhance MPC using neural network–based predictive modeling to improve robustness [2]. Unlike these approaches, we employ monocular pose estimation via Perspective-n-Point (PnP) for a lightweight open-loop docking controller. For navigation, we extend ROS2 Navigation2 [5] with multi-phase control tailored to indoor delivery tasks. Our design also connects to recent vision-language navigation systems such as LM-Nav [7] and VLMaps [3], which demonstrate how large pre-trained models can ground natural-language instructions in spatial environments. At the intent-parsing level, our Physical AI module draws inspiration from LLM-driven robotics frameworks such as ChatGPT for Robotics [8], SayCan [1], and CoPAL [4], using GPT-4o-mini to transform speech into structured, verifiable commands that trigger a topic-gated finite-state machine.

This paper presents the design and deployment of an end-to-end voice-controlled robotic delivery system. Our central contribution is a modular software architecture that reliably bridges natural-language commands and ROS2 execution by combining a lightweight speech-to-command pipeline integrating ASR and LLM-based parsing with a topic-gated orchestration engine that

coordinates all subsystems. We validate the platform through repeated real-world trials, demonstrating reliable end-to-end delivery despite challenges such as perception latency, CPU load, LiDAR occlusion, and multi-task integration.

2 Methods

Our system uses a voice-controlled indoor delivery robot that combines natural-language commands with autonomous perception, manipulation, and navigation. User speech is transcribed and parsed by an LLM-based Physical AI module, then passed to a centralized orchestration engine that sequences actions and manages subsystem readiness. The perception module detects objects and provides 3D positions for the manipulation arm to pick and load items, after which the navigation stack guides the robot using ROS2 Nav2 and a lightweight docking controller. Throughout the process, the orchestrator synchronizes modules and provides real-time status updates, enabling end-to-end delivery from a single spoken command. The workflow of the system is shown in Fig. 1.

2.1 Physical AI and Robot Orchestration

The Physical AI module enables natural-language interaction by converting user speech into structured robot commands. As shown in Fig. 2, audio from the microphone is streamed to the GPT-4o-mini ASR model, and the resulting transcript—combined with real-time telemetry from navigation, perception, and manipulation—is passed to GPT-4o for intent recognition. The model receives the transcript, the robot's current position and goal information, the latest 3D object list, and the arm's operational status. A constrained JSON schema ensures that the output is either a validated command or a request for clarification. Once a command is approved, the module publishes it to the orchestration layer and simultaneously listens to system feedback topics to generate spoken and visual status updates for the user interface. This interface also includes a real-time map visualization, where precomputed affine transformations allow the robot's position to be updated at approximately 10,Hz.

The Orchestration module serves as the system's centralized decision-making engine, coordinating all subsystems—Physical AI, Perception, Manipulation, and Navigation—through a structured finite state machine (FSM). Upon receiving a parsed command, the orchestrator advances the task through stages, including intent parsing, object detection, pickup, navigation, delivery, and completion. Each stage checks subsystem readiness, dispatches appropriate actions, and monitors feedback from navigation and manipulation nodes to determine whether to proceed, retry, or abort. All state transitions are logged persistently, enabling traceability and recovery across sessions. Throughout execution, the orchestrator verifies object availability, schedules navigation goals, supervises grasp and transfer actions, and synchronizes auxiliary behaviors such as lid control and docking of the mobile robot. A persistent task queue regulates multi-command operations, ensuring serial execution and safe recovery from failures such as missed grasps or navigation timeouts. Real-time status updates are broadcast to the UI and Physical AI modules, supporting transparent monitoring and fast debugging across the integrated system.

2.2 Navigation and Mobile Robot

The mobile robotic platform uses a modular ROS 2 architecture integrating a Create2 mobile base, RPLIDAR A1, OAK-D RGB-D camera, and a servo-actuated lid to support autonomous navigation, perception, and object handling. Core subsystems communicate through a minimal topic set—including lidar, camera, and controls —allowing clean separation between sensing, control, and actuation layers. A unified ROS launch file synchronizes all nodes, enabling consistent startup behavior, namespace isolation, and seamless integration with downstream navigation and docking modules. A custom 3D-printed container consolidates the LiDAR mount, OAK-D housing, servo cutout, and internal compartments for compute hardware and batteries. Hardware design also enhances mechanical stability, simplifies wiring, and resolves issues with servo misalignment and Jetson Nano thermal dissipation.

To ensure reliable operation and extended runtime, the system employs a dual-power configuration with isolated supply lines for compute and actuation, preventing brownouts and improving debugging safety. Network communication across the Jetson Nano and orchestration workstation is managed through a shared `ROS_DOMAIN_ID`, while bandwidth optimizations—including JPEG compression of camera streams for better frame rate.

The navigation of the mobile robot uses ROS 2 Nav2 with LiDAR-based localization and a merged SLAM map to perform point-to-point navigation within the lab. ArUco-based pose estimation provides final docking alignment. A status publisher reports position, distance-to-goal, ETA, and load-readiness. The robot's container lid is controlled via topic commands, closing before navigation and reopening after docking.

We generated two partial maps (kitchen and computer lab area) using Cartographer and merged them in GIMP, performing alignment and manual cleanup to ensure Nav2's global costmap interpreted obstacles correctly (Fig. 3).

Navigation with Nav2: A centralized ROS 2 loop (*Destination-Client*) polls for delivery tasks and executes a fixed sequence: lid actuation, retreat, Nav2-based navigation, return to a pre-docking pose, and open-loop docking. Navigation goals (x, y, θ) are sent to *NavigateToPose*, with orientation converted via

$$q_z = \sin(\theta/2), \qquad q_w = \cos(\theta/2), \qquad q_x = q_y = 0.$$

State updates are published before and after each cycle for system-level coordination.

Marker-Based Docking: Docking uses monocular ArUco tracking. Marker pose is obtained via OpenCV PnP:

$$\mathbf{T}_{\text{marker}}^{\text{camera}} = \begin{bmatrix} \text{Rodrigues}(\mathbf{r}) & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad \mathbf{T}_{\text{marker}}^{\text{base}} = \mathbf{T}_{\text{camera}}^{\text{base}}\, \mathbf{T}_{\text{marker}}^{\text{camera}}.$$

From (x, y) in the base frame, the robot computes heading error $\phi = \tan^{-1}(y/x)$, rotates for $t_r = |\phi|/\omega$, then drives forward for

$$t_f = \frac{\sqrt{x^2 + y^2} + \delta}{v},$$

using linear velocity v, angular velocity ω, and buffer δ for network-delay robustness.

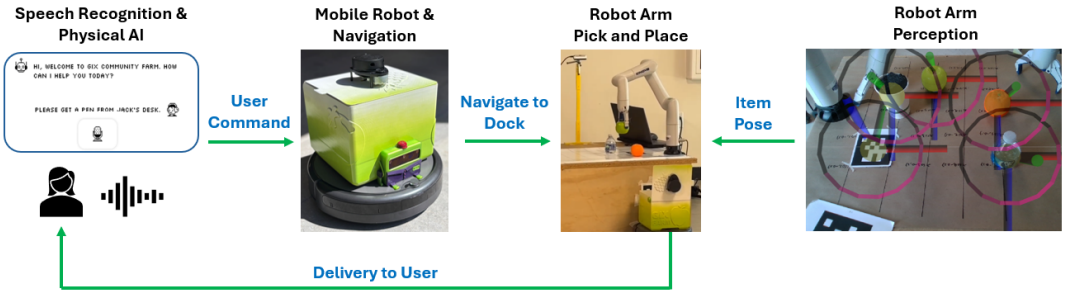

Figure 1: Workflow of the system. The procedure is fully automated, and the users can speak as input and receive voice feedback. The system is built and tested with real robots.

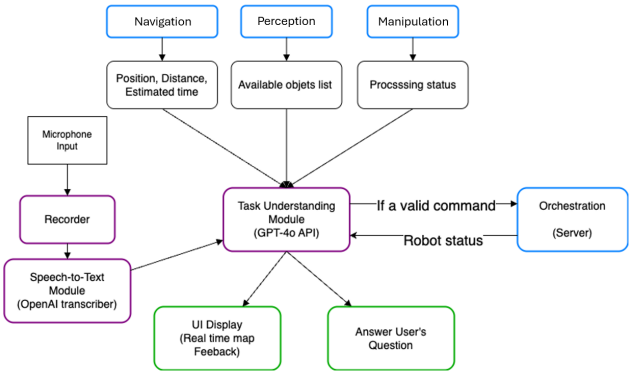

Figure 2: Physical AI architecture integrated with subsystem ROS infrastructure.

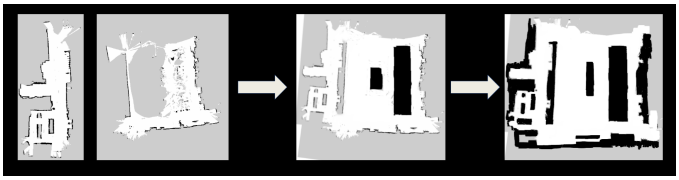

Figure 3: Map building: (1) Kitchen map, (2) Lab map, (3) Initial merge, (4) Cleaned final map.

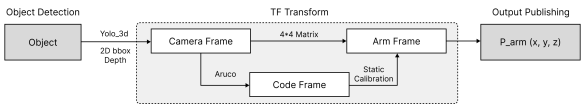

Figure 4: The architecture of the perception subsystem.

2.3 Robot Arm Manipulation and Perception

The perception and manipulation modules work together to detect objects, localize them relative to the robot arm, and execute reliable pick-and-place actions that support the overall delivery workflow. Perception provides 3D object positions and scene awareness, while manipulation transforms these observations into physical interaction through a Kinova Gen3 Lite arm and gripper.

The perception pipeline (Fig. 4) uses a RealSense RGB-D camera feeding both a YOLOv8 detector for 3D object bounding boxes and an ArUco-based localization module. We use an open-source YOLOv8 model to detect common delivery items (e.g., fruits, cups, and bottles). A fixed marker near the arm establishes the camera–arm transform, enabling detected objects to be projected into the arm's base frame. YOLO produces 2D and 3D detections, which are depth-corrected and transformed via a calibrated matrix to yield pose estimates that are published continuously to the rest of the system. These messages allow the orchestrator and manipulation module to confirm object availability, validate user commands, and plan safe grasps. The perception stack operates reliably at real-time rates (10–15 Hz), achieving desirable detection accuracy under normal lighting and table configurations.

Building on these inputs, the manipulation module executes autonomous pick-and-place behaviors using a state-machine architecture layered on MoveIt2. Upon receiving a task goal from orchestration, the arm transitions through stages including target approach, grasp execution, object transfer, release, and return-to-home. Cartesian plans with conservative velocity and acceleration constraints ensure smooth motion near objects. Grasp detection is performed using gripper-position feedback, where deviations from the open-state baseline indicate successful contact, and a dual-threshold mechanism enables early detection of drops. Safety is further supported through force monitoring of joint efforts, enabling rapid emergency stops when collisions or excessive loads are detected. In such cases, motion is halted, the arm retreats to a safe height, and the system enters a temporary lockout before resetting automatically.

Together, the perception and manipulation modules are intended to create a robust sensing-to-action pipeline: objects are detected and localized in real time, transformed into actionable coordinates, evaluated for graspability, and then physically retrieved and placed into the robot's container. This integrated design enables reliable object handling within the broader voice-driven delivery system and supports coordinated behavior across all subsystems.

3 Experiments and Results

To validate the performance of the proposed robotic system in real-world applications, experiments were conducted with real robot hardware (introduced in Section 2) in a controlled indoor lab

Table 1: Preliminary Subsystem Performance Metrics

Subsystem	Metric	Result
HW/SW	Lid actuation success	[39/42, 93%]
	Jetson stability	Stable
Orchestration	Avg mission time	[10 min]
Navigation	Path completion	[95%]
	Docking success	[14/20, 70%]
	Docking offset	[5 cm]
Manipulation	Grasp success	[25/27, 93%]
	Trajectory latency	[2–5 s]
Perception	Classification accuracy	[100%]
Physical AI	STT accuracy	[25/25, 100%]
	Intent parsing	[25/25, 100%]

environment. Each full-cycle run involved a human user querying available objects, issuing a spoken command, parsing and dispatching the task through the Physical AI and orchestration layers, navigating to the pickup station, performing object loading via manipulation and lid control, navigating to the drop-off station with AprilTag docking, and completing handoff to the user. Subsystems were also tested independently throughout development, with a full-system trial completing successfully. These tests allowed the subsystems to validate core functionality while identifying integration issues related to perception latency, network stability, and hardware dependencies.

Table 1 summarizes subsystem-level metrics collected during development, including high object-classification accuracy, reliable speech-processing performance, and strong navigation results. In the successful end-to-end run, the robot completed the full mission—from spoken request to delivery and return—in 8 minutes and 46 seconds, demonstrating interoperability across all modules.

4 Conclusion

This paper presents an integrated indoor delivery robot that translates spoken requests into multi-step mobile manipulation using a ROS2 orchestration framework. A finite-state-machine controller bridges high-level human intent and low-level subsystem actions, enabling end-to-end delivery missions from spoken request to completion. Real-world trials demonstrated the system's reliability, including an 8 min 46 s successful run with high navigation accuracy, low docking error, and desirable object classification. The project also revealed practical integration lessons involving message contracts, network bandwidth, and transparent system feedback. Future work will expand data collection, add closed-loop visual servoing, move perception processing fully on-board, and incorporate more advanced recovery and scheduling strategies.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[2] Seong Hyeon Hong, Benjamin Albia, Tristan Kyzer, Jackson Cornelius, Eric R Mark, Asha J Hall, and Yi Wang. 2024. Artificial neural network-based model predictive visual servoing for mobile robots. *Robotica* 42, 8 (2024), 2825–2847.

[3] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. 2023. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 10608–10615.

[4] Frank Joublin, Antonello Ceravola, Pavel Smirnov, Felix Ocker, Joerg Deigmoeller, Anna Belardinelli, Chao Wang, Stephan Hasler, Daniel Tanneberg, and Michael Gienger. 2024. Copal: corrective planning of robot actions with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 8664–8670.

[5] Steve Macenski, Francisco Martín, Ruffin White, and Jonatan Ginés Clavero. 2020. The marathon 2: A navigation system. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2718–2725.

[6] Roni Permana Saputra, Midriem Mirdanies, Eko Joni Pristianto, and Dayat Kurniawan. 2023. Autonomous Docking Method via Non-linear Model Predictive Control. In *2023 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*. IEEE, 331–336.

[7] Dhruv Shah, Błażej Osiński, Sergey Levine, et al. 2023. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*. PMLR, 492–504.

[8] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. 2023. Chatgpt for robotics: Design principles and model abilities. 2023. *Published by Microsoft* (2023).

Received —-; revised —-; accepted —-