Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Hybrid Visual Perception and LLM Framework for Service Robot Planning and Navigation

**Ahmad Ataka**

ahmad.ataka.ar@ugm.ac.id

Universitas Gadjah Mada

**Yanayir Rifai**

Universitas Gadjah Mada

**Yusuf Badriawan**

Universitas Gadjah Mada

**Burhanudin Burhanudin**

Universitas Gadjah Mada

**Igi Ardiyanto**

Universitas Gadjah Mada

**Jun Miura**

Toyohashi University of Technology

**Dino Accoto**

FlandersMake@KULeuven

**Research Article**

**Keywords:** service robots, large language models, visual perception, task planning, autonomous navigation, embedded robotics

**Posted Date:** November 12th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11370496/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

**Additional Declarations:** No competing interests reported.

# Hybrid Visual Perception and LLM Framework for Service Robot Planning and Navigation

Ahmad Ataka[1*], Yanayir Rifai[1], Yusuf Badriawan[1], Burhanudin[1], Igi Ardiyanto[1], Jun Miura[2], Dino Accoto[3,4]

[1*]Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, 55281, Indonesia.
[2]Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Aichi, Japan.
[3]FlandersMake@KULeuven, Celestijnenlaan 300, Leuven, 3000, Belgium.
[4]Robotics, Automation and Mechatronics, Department of Mechanical Engineering, KU Leuven, Ghent, 9000, Belgium.

*Corresponding author(s). E-mail(s): ahmad.ataka.ar@ugm.ac.id;

**Abstract**

In the last decade, service robots have gained significant attention for their role in supporting humans across various applications. To operate effectively and adaptively, service robots require environmental perception, natural-language understanding, and planning and navigation capabilities. However, traditional planning frameworks struggle to provide the required flexibility. In contrast, end-to-end multimodal systems based on vision-language models (VLMs) require substantial computational resources, making them unsuitable for deployment on embedded robotic platforms. This paper proposes a hybrid framework that combines fast, deep-learning-based visual perception with the reasoning capabilities of a large language model (LLM) to enhance task comprehension, planning, and navigation for service robots. By decoupling vision from language reasoning, we avoid the complexity of a unified vision-language model, allowing the proposed framework to be deployed on a lightweight embedded platform with substantially lower latency than the VLM baseline. Experimental results show a 96% task completion rate in structured mission tests and 84% overall success in hybrid perception tasks. Moreover, the proposed approach achieves 3× to 20× faster inference than VLM-based baselines under both API-based and local deployment settings. Real-robot deployment confirmed the successful transfer of the full perception-to-action pipeline to the physical platform.

**Keywords:** service robots, large language models, visual perception, task planning, autonomous navigation, embedded robotics

## 1 Introduction

Service robots have gained significant attention in the last decade. Robots, initially designed for static, well-defined factory-floor environments, have now moved into unknown and dynamic settings such as restaurants [1], hospitals [2], and agriculture [3]. Consequently, robots are now expected not only to perform tasks accurately, but also to perceive their surroundings rapidly, adapt reactively to dynamic situations, and interact

actively with humans. Although robot planning and navigation approaches have been thoroughly explored [4], challenges remain in integrating human-instruction understanding, task planning, environmental perception, and task execution into a unified process.

Advances in artificial intelligence have significantly improved robot performance over the last five years. On the one hand, deep-learning approaches have substantially enhanced robots' visual perception capabilities [5]. These advances are particularly useful for understanding dynamic environments and improving robot decision-making. On the other hand, the emergence of large language models (LLMs) has revolutionized human-robot interaction. Inspired by LLM-powered systems such as ChatGPT, roboticists have rapidly adopted the natural-language capabilities of LLMs to improve robots' understanding of human instructions and task planning [6]. With the ability to perceive the environment and understand human instructions, robots can potentially perform more challenging service tasks. However, challenges remain in balancing performance, computational complexity, and execution time. This requirement is particularly relevant for guide-service robots in lobbies, hallways, and reception areas, where onboard computing and sensing resources are often limited.

Several studies have used LLMs to improve robot navigation. One study employed an LLM in a mobile-robot navigation setting for scene understanding [7]. Another study combined language and sketching through an LLM to enhance robot navigation [8]. LLM-based zero-shot object navigation has also been proposed [9]. LLM-based methods have additionally been proposed for mobile-robot trajectory generation [10] and manipulation [11]. However, these studies have not integrated visual perception, a critical component of service robots, into planning and task execution.

Other approaches use LLMs not only for language understanding but also for visual and sensory interpretation. One proposed solution focuses on landmark understanding by combining a Vision Navigation Model (VNM), a VLM, and an LLM to build a topological graph of the environment, identify landmark nodes, and then run a navigation policy that follows the selected path [12]. Similar work has used an LLM and a VLM for an armed quadruped robot to ground spatiotemporal navigation commands using landmarks. However, these approaches rely on landmarks placed in the environment, which can make them less practical [13]. Another approach, Agri-VLN (Vision-and-Language Navigation), develops an end-to-end solution from navigation commands to actions in agricultural settings. It can decompose tasks into subtasks, but it allows only simple movements without complete map-based navigation [14]. A similar approach, Nav-GPT2, which serves as a bridge between specialized VLN and LLM-powered navigation, also suffers from the same limitation [15]. Recent work has also employed LLMs to transform visual and sensory data into navigation commands [16]. Taken together, these studies demonstrate strong semantic capabilities, but they either depend on environmental landmarks, do not support full map-based service navigation, or incur substantial computational costs.

A practically relevant direction is to combine lightweight visual perception with language-based reasoning for embedded robotic deployment. A recent study targeting dense environments with pedestrians employed YOLO for visual perception and an LLM for task understanding and high-level planning [17]. However, because it relies on numerous sensors, it remains computationally demanding. LLMs and YOLO have also been employed for a robotic arm [18]. However, to the best of our knowledge, existing studies do not clearly demonstrate a lightweight, decoupled perception-LLM-navigation pipeline for an embedded mobile service robot, evaluated on hybrid perception tasks in terms of success rate and inference latency.

Building on our prior work on VLM-assisted spatial goal reasoning for guide-service robot navigation [19], this paper develops a decoupled YOLO–LLM framework for task comprehension, planning, and navigation in service robots. Whereas the prior study focused on using spatial context to infer navigation goals, the present work adds task-relevant visual perception, structured $M$-, $S$-, and $V$-type task execution, and an end-to-end evaluation of task success and inference latency. By decoupling vision from language reasoning, we avoid the complexity of a unified vision-language model, allowing the proposed framework to be deployed on a lightweight embedded platform with lower inference latency than unified

VLM-based alternatives. We demonstrate its performance in real-world experiments, showing that the robot can interpret natural-language instructions, integrate visual perception results into the planning step, and execute the resulting tasks successfully. To the best of our knowledge, this is the first study to systematically demonstrate a decoupled, lightweight perception-LLM framework for mobile-robot navigation that achieves both a high task-success rate and real-time feasibility on embedded robotic platforms.

Experimental results show that the system achieves a 96% success rate in structured mission tests and an 84% success rate in hybrid perception tests. Compared with the VLM baseline, the proposed system delivers approximately 3× to 20× faster inference, highlighting its suitability for real-time robotic applications on resource-constrained platforms.

## 2 Service Robot Platform

The guide-service robot used in this study is built on a TurtleBot3 base, which provides a modular, ROS-native foundation for system development and integration. To improve ergonomic accessibility and interaction range, a 1.2-m chassis extension was installed, enabling more comfortable human-robot interaction in public-service environments.

The Jetson Orin Nano serves as the primary embedded computing unit and was selected to validate deployment under the resource-constrained conditions typical of lightweight robotic platforms. To support natural human-robot interaction, additional capabilities were integrated, including wake-word detection, automatic speech recognition, speech-to-text, and text-to-speech pipelines. These components collectively support language-driven task execution and mission planning. A standard monocular camera is employed as the sole visual-perception modality, reinforcing the lightweight and computationally efficient system design. Consequently, the robot supports both touchscreen-based interaction and voice-command execution, providing a multimodal interaction interface suitable for service-oriented robotic deployment.

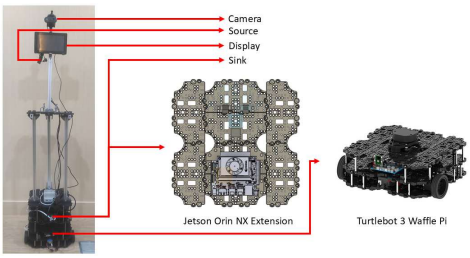



**Fig. 1**: Components of the guide-service robot used in the experiments.

Figure 1 shows the robot and its components. Overall, the service robot was intentionally designed to operate within the computing and power constraints of embedded robotic hardware. The framework is portable to other mobile robots, provided that they offer monocular visual input, localization on a 2D map, and a navigation stack capable of receiving symbolic or waypoint-level targets.

## 3 A Hybrid Visual Perception and LLM Framework

This section describes the proposed hybrid framework for service-robot task execution. To support flexible service-robot behavior under limited onboard resources, the proposed system is organized as a modular hybrid framework rather than a monolithic pipeline. This design separates command interpretation, perception, planning, and execution, while allowing perceptual feedback to refine or redirect ongoing actions.

The proposed framework accepts three types of input: a natural-language command, a prebuilt map, and live visual data. It then outputs structured tasks and executable robot actions. To achieve this, the framework is organized into a mission-planning module and an execution module.

### 3.1 Mission Planning Module

The mission-planning module translates high-level commands into structured executable plans, as illustrated in Fig. 2. Each natural-language command is converted into a task sequence $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$, where each $t_i$ specifies one of the following action types:

- Move ($M$): a robot-movement or destination-selection task, such as "(move, front office)";
- Vision ($V$): an object-finding task, such as "(vision, people, True)"; and
- Speak ($S$): a spoken-response-generation task, such as "(speak, hello)".

Among these components, the vision action is treated as a special case. It may include an additional Boolean value indicating whether the visual process should be executed simultaneously with other actions ("True") or independently ("False"). For instance, the task "(vision, people, True)" indicates that the object-finding task is executed simultaneously with other tasks, corresponding to the input instruction "on your way to the bathroom." By contrast, the task "(vision, people, False)" indicates that the object-finding task is executed after the previous task has been completed, corresponding to the input instruction "and find a person." Furthermore, the mission planner operates modularly, allowing visual perception to be dynamically incorporated or excluded based on task requirements.

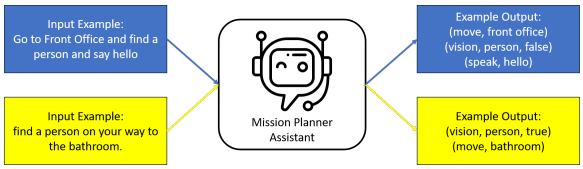



**Fig. 2**: The mission-planning module translates a natural-language command into a structured executable plan.

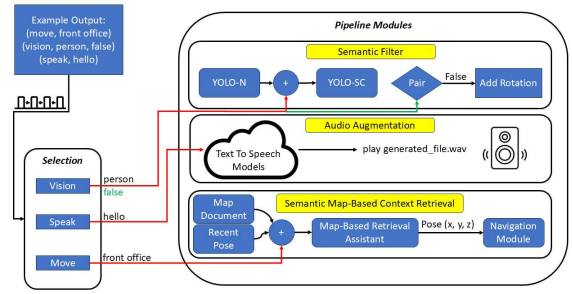



**Fig. 3**: The task-based selection module accommodates different command types, namely Move $M$, Speak $S$, and Vision $V$, and produces the corresponding actions.

## 3.2 Execution Mechanisms

The mission plan $\mathcal{T}$ generated by the mission-planning module is then executed through a customized branching pipeline, as shown in Fig. 3. During execution, the execution-mechanism module runs all components of the mission plan $\mathcal{T}$ sequentially in their assigned order. The pipeline branches accommodate three task types: Move ($M$), Vision ($V$), and Speech ($S$). We employ semantic map–based context retrieval for the $M$-type task, a semantic-filtering mechanism for the $V$-type task, and audio augmentation for the $S$-type task.

### 3.2.1 Semantic Map-Based Context Retrieval

To interpret high-level and intuitive commands (such as "Where is the nearest bathroom?") correctly, the system must reason over location-aware contextual information. Key elements required for this process include the number of available target locations, their respective spatial coordinates, and the robot's current position. To address this requirement, we employ an agent capable of performing document-based searches.

In our approach, the robot's coordinates and the target-location coordinates are provided in a map document as part of the prompt whenever an $M$-type task is assigned within the mission pipeline. The OpenAI file-search capability enables the model to reason over these coordinate embeddings. The coordinates are obtained from manually indexed map entries generated after a mapping session, allowing the LLM to reason semantically about the environment's spatial

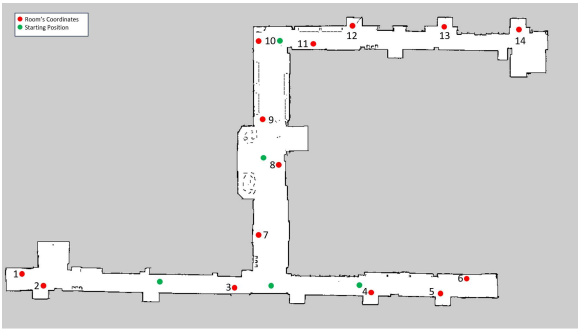



**Fig. 4**: Example of the environment map used as contextual information during context retrieval.

structure. For instance, if a user asks the robot to go to a specific location relative to its current position (such as the "nearest bathroom"), the LLM combines information about the room label, the topological map context, and the current robot pose to determine the correct destination.

An example of the map information provided to the agent to determine the correct destination is shown in Fig. 4. This process produces a pose $(x, y, z)$ that serves as the robot's target position. The resulting pose is then sent to the low-level navigation module, which uses this information together with the prerecorded map to generate motor commands for the robot.

### 3.2.2 Semantic Filter

The semantic filter uses the target specified in each $V$-type task to guide the visual-perception stage through the YOLO detector. We define two operating modes. The first is a neutral configuration, denoted as YOLO-N, in which YOLO operates without semantic restriction. In this mode, detection is performed over the predefined object classes available to the system. The second is a semantically conditioned configuration, denoted as YOLO-SC, in which detection is restricted to the task-relevant classes inferred from the current command. This conditioning reduces unnecessary detections and focuses perception on objects that are relevant to the assigned task.

In addition, each $V$-type task includes a Boolean attribute, *pair_exist*, that indicates whether the visual subtask is coupled with an $M$-type movement task. If *pair_exist* is true, visual perception is executed in conjunction with the associated movement task. If it is false, the perception task is treated as independent. In that case, the robot performs an active exploratory rotation to expand its field of view and improve detection coverage. In this work, this mechanism is mainly used for object-finding scenarios, which form the primary vision task considered in the proposed framework.

### 3.2.3 Audio Augmentation

Audio augmentation is implemented to generate spoken responses from textual input using text-to-speech models. For each $S$-type module, the associated object is passed as an argument to the speech-generation pipeline. The system invokes the text-to-speech API to synthesize a waveform audio file in WAV format based on the provided content, which is then played through the robot's onboard audio output. This mechanism enables natural and context-aware verbal interaction between the robot and users. The audio module is used solely for user interaction and does not alter the mission-planning logic.

## 4 Experimental Setup

We evaluate the proposed framework through four complementary scenarios: an $M$-type command test and a full command test in Gazebo simulation, a single-inference latency comparison, and a real-robot deployment test. Together, these scenarios assess semantic-map reasoning, end-to-end perception–action execution, computational responsiveness, and physical deployment feasibility. Unless stated otherwise, a trial is counted as successful only when the system identifies the intended semantic target, generates the appropriate executable action, and completes the assigned task under the evaluation rule defined for that test.

### 4.1 $M$-type command test

The first experiment, denoted as the $M$-type command test, assesses the accuracy of missions whose instructions consist solely of motion-based actions. This test isolates language-guided target selection and semantic-map reasoning without visual perception, thereby establishing whether failures in later tests arise from perception coupling rather than from the map-based planning stack itself. Each experiment begins from one of several predefined starting points. To test the robustness of the system, we designed five natural-language command prompts and executed each prompt from five distinct starting locations. The commands are as follows:

1. "Go to the nearest room."
2. "Go to the two nearest rooms."
3. "Go to the two nearest laboratories."
4. "Go to the Front Office, then navigate to the nearest laboratory."
5. "Go to the nearest bathroom, then proceed to the next room."

### 4.2 Full Command Test (Move $M$, Speak $S$, and Vision $V$)

The second experiment is referred to as the full command test. This task evaluates the performance of missions that integrate motion, speech, and visual perception, thereby highlighting the system's ability to operate with the vision module enabled. This test uses the same experimental configurations as the first test. In addition to movement commands, we evaluated visual perception by placing multiple objects along the robot's trajectory for detection by the vision module. The evaluation considers three object classes: $x$ denotes a fire hydrant, $y$ denotes a chair, and $z$ denotes a person. These classes were selected to represent scenarios applicable to both simulation and real-world deployments. The natural-language commands for this task are provided below:

1. Go to front office and look for a fire hydrant.
2. Find a chair in the nearest lab.
3. Is there any person in the nearest room?
4. Please go to the front office and check if someone is there.
5. Please look for anybody available to help while you are on your way to the bathroom.

### 4.3 Single Inference Test

In the third experiment, we compare visual inference latency between a single-frame YOLO-SC pipeline and a single-frame VLM setup using single-hit API calls. Response time is used as the primary metric to demonstrate that the proposed modular design achieves faster inference, indicating lower computational demand and improved suitability for real-time robotic deployment.

In the VLM-based approach, the prompt and image are provided directly to the model through its standard inference pipeline. In contrast, the proposed hybrid visual-LLM approach processes the user command through an initial reasoning step to generate an enumerated object class compatible with the YOLO detector. Importantly, this reasoning step is executed only once per command. This setup captures the initial response latency, including comparable preprocessing and inference stages. By isolating the first inference cycle, the test highlights the response-time characteristics of each method while avoiding bias from repeated or cached reasoning steps.

### 4.4 Real Robot Test

Finally, real-robot experiments validate the end-to-end performance of the proposed system on the fully assembled physical platform. The physical experiment evaluates the same command-interpretation, semantic-filtering, navigation, and action branches used in simulation. The framework was deployed without structural modification. A successful physical trial requires the robot to interpret the command, navigate to the intended semantic target, and execute the corresponding perception or interaction action. The tests were performed in the first-floor lobby environment depicted in Fig. 4.

## 5 Results and Analysis

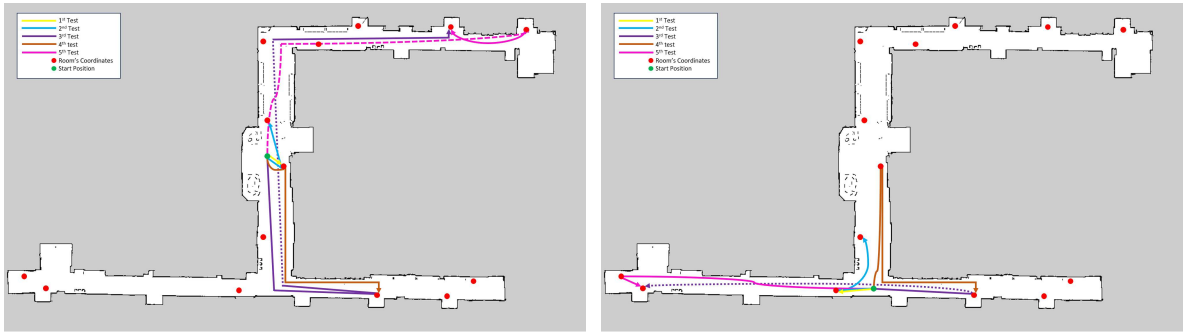
This section presents and analyzes the experimental results from the four scenarios described in the previous section. The results are reported in four parts: mission-only reasoning, full command execution, real-robot transfer, and inference-time comparison. Together, these analyses evaluate semantic-planning accuracy, end-to-end task completion, physical-deployment feasibility, and computational efficiency. The results are reported using both quantitative and qualitative measures. Quantitatively, performance is summarized using several metrics to facilitate direct comparisons across test cases. Qualitatively, additional figures illustrate execution details, such as the generated navigation paths and task outcomes.

### 5.1 $M$-type command test

To evaluate the effectiveness of the $M$-type task execution, we conducted experiments using five predefined starting positions and five distinct mission commands, resulting in a total of 25 individual test runs. Each test assesses the robot's

**Table 1**: Performance summary

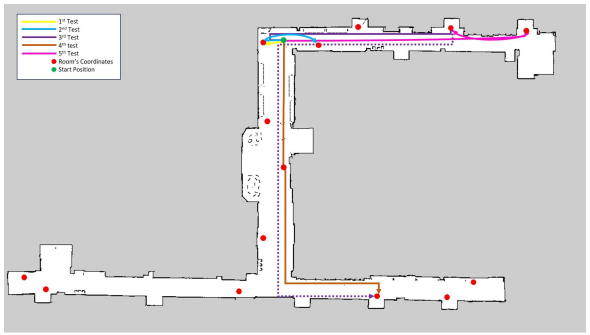
| Metric | Mission planning | Task accuracy |
|---|---|---|
| Score | 100% | 96% |



(a) Position 1



(b) Position 2



(c) Position 3



(d) Position 4



(e) Position 5

**Fig. 5**: Resulting paths for the $M$-type command test. The paths are manually reconstructed to illustrate inferred destination selection and do not represent the exact executed trajectories.

ability to correctly interpret spatial commands and generate an appropriate navigation target.

We use two metrics to assess performance quantitatively. The first is mission-planning accuracy, which refers to the ability of the mission-planning module to interpret and decompose user instructions correctly. This metric evaluates whether the system can transform a given natural-language command into a properly structured and ordered list of executable subtasks according to the predefined pipeline design. The second metric is task accuracy, which measures execution-level performance. It evaluates whether the robot successfully completes the planned commands from five distinct starting positions. This metric reflects the robustness of navigation and action execution

under varying spatial conditions. Thus, mission-planning accuracy assesses the correctness of reasoning and task decomposition, whereas task accuracy measures the successful simulated execution of the planned actions.

The results are summarized in Table 1. In our evaluation, the mission-planning module achieved 100% accuracy, indicating that no logical or structural errors occurred during instruction decomposition across all tested scenarios. Task accuracy was 96%, indicating that the robot successfully completed most of the planned commands from five distinct starting positions. These results indicate that the semantic-map retrieval and mission-planning components are reliable when visual grounding is not required.

The results are also visualized in Fig. 5, which illustrates the destination points produced by the semantic map–based context-retrieval module. The navigation paths shown in the figure were manually reconstructed for clarity because of overlay limitations in the simulation interface. Thus, the reconstructed lines do not represent the exact trajectories executed by the robot; rather, they indicate the inferred navigation directions for evaluation purposes.

In the task-accuracy evaluation, the model occasionally failed to select the correct destination. As illustrated in Fig. 5d, the fifth command scenario repeatedly produced the same incorrect destination for a multi-step command. The failure occurred during semantic-map reasoning. Because the same incorrect output was reproduced across attempts, this result suggests a limitation in the prompt or contextual map representation rather than a navigation failure.

Another performance limitation is shown in Fig. 5e, where a contextual conflict arises in the spatial interpretation. Based solely on coordinate-based Euclidean distance, the system calculates the nearest bathroom using geometric proximity as the primary metric. Under this formulation, the bathroom at the top-right should be selected as the closest destination. However, the model selected the bottom-left bathroom, which corresponds to a shorter navigation path in the actual layout. Although this selection deviates from strict coordinate-based proximity, the resulting path remains operationally valid and acceptable in practice.

This observation reveals that conflicting contextual interpretations may arise between geometric distance and navigational feasibility. In this work, coordinate-based reasoning serves as the formal decision basis; however, in cases where the executed path remains logically consistent and functionally correct, such as in Fig. 5e, the outcome is considered a successful task under a path-validity criterion. This dual interpretation explains the acceptance of specific edge cases while maintaining overall evaluation consistency.

## 5.2 Full Command Test (Move $M$, Speak $S$, and Vision $V$)

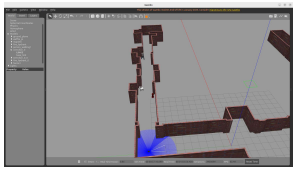
The full command test was conducted using the same experimental configuration as the M-type evaluation, namely five distinct command prompts executed from five predefined starting positions, resulting in a total of 25 test cases. In contrast to the M-type test, this evaluation primarily emphasizes the system's vision-driven semantic filtering and action-triggering capability.

Under the defined test conditions, the proposed framework achieved an overall success rate of 84%. Although this success rate was lower than that of the $M$-type test, no additional parameter tuning or controller refinement was performed for this experiment. The navigation strategy relies exclusively on monocular object detection, using two-dimensional bounding-box information without depth estimation. As a result, precise final alignment—such as an optimal frontal orientation toward the target—cannot be consistently guaranteed. Minor deviations in pose or heading were therefore classified as failures under the strict evaluation metric. However, from a functional standpoint, all trials successfully reached their intended semantic targets, as illustrated in Fig. 6.

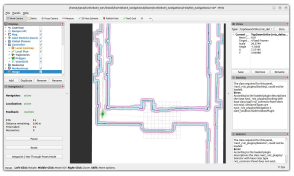
These results highlight the inherent limitations of monocular perception, particularly in scenarios where objects dominate the field of view due to close proximity or appear at smaller scales with lower detection confidence. Nevertheless, the perception-to-action pipeline remained operational across all cases, demonstrating the robustness of the proposed semantic filtering framework under constrained visual conditions.

## 5.3 Real Robot Test

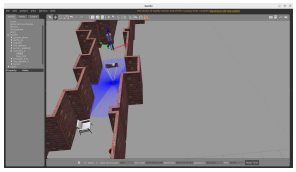
To further validate practical applicability, a sim-to-real transfer was conducted for one test (Test 4) in the full-command experiment. The complete execution pipeline—from command interpretation and semantic filtering to navigation and perception—was deployed on the physical robot platform, as illustrated in Fig. 7. The system was executed without structural modification, using the same integrated framework implemented in simulation. During real-world deployment, the robot operated under actual environmental conditions and in the presence of people, performing the assigned task reliably and stably. Overall,



(a) Initial robot configuration



(b) Map-based navigation toward the target

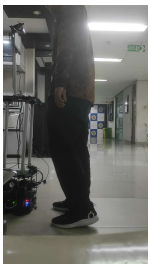


(c) Robot arrival at the target location

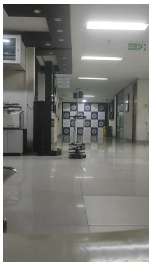



(d) YOLO-SC person detection during execution

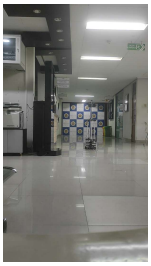
**Fig. 6**: Representative full-command experiment: the robot navigates toward the bathroom while performing perception-based person detection.



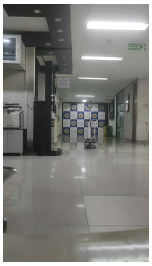
(a) User command interaction



(b) $M$-type navigation execution



(c) $V$-type semantic-filtered perception (YOLO-SC)



(d) Active exploratory rotation for object search

**Fig. 7**: Real-world execution of the proposed hybrid perception–LLM framework on the guide-service robot. The robot receives a user command, navigates to the semantic target, and performs task-relevant visual perception.

the results demonstrate that the proposed hybrid pipeline transfers effectively from simulation to physical implementation while maintaining consistent operational performance. Despite the limited number of physical trials, the real-robot tests confirm that the proposed pipeline operates reliably in a real-world environment. The observed behavior is consistent with the simulation results, demonstrating successful task execution and supporting the feasibility of deployment on a physical robotic platform.

## 5.4 Single Inference Test

We conducted a comparative inference evaluation using the same detection configuration across two architectural settings: a VLM-based pipeline and the proposed hybrid framework (LLM + YOLO). The experiments were performed under two deployment modes: (1) fully local inference

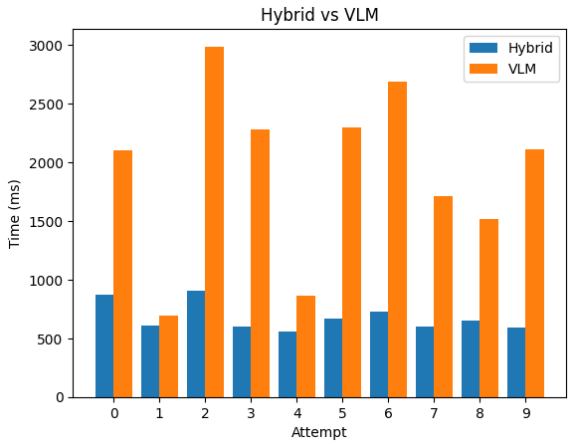



**Fig. 8**: Response-time comparison between the VLM and hybrid frameworks.

and (2) API-based inference for both approaches. Latency was recorded for each configuration to ensure a fair comparison under equivalent conditions.

The results consistently show that the hybrid framework outperforms the VLM in all tested scenarios. In the API-based setting, as illustrated in Fig. 8, the VLM exhibits approximately three times the latency of the hybrid framework. The performance gap becomes even more pronounced under fully local deployment, where the measured average inference time of the VLM is approximately 20 times that of the hybrid approach, as summarized in Table 2.

These findings demonstrate that the proposed LLM–YOLO architecture is better suited to lightweight embedded deployment, offering substantially lower latency while maintaining functional performance within the designed experimental setup. The VLM offers broader, open-ended semantic flexibility, whereas the hybrid YOLO–LLM pipeline sacrifices some generality in exchange for lower latency and lower deployment costs on embedded hardware.

**Table 2**: Average response-time comparison

| Method | VLM | Hybrid |
|---|---|---|
| Local | 9650 ms | 439.466 ms |
| Cloud | 1924.11 ms | 677.74 ms |

# 6 Conclusion

This paper presents a hybrid perception–reasoning framework designed for lightweight configuration and efficient embedded deployment. By decoupling semantic reasoning from visual inference and structuring the system into a modular task pipeline, the proposed approach enables robust mission execution while maintaining low computational overhead. Experimental validation shows that the proposed decoupled framework achieves 96% success in map-based mission tests, 84% success in hybrid perception-driven tests (primarily affected by pose-alignment constraints rather than semantic failure), and 3× to 20× lower inference latency than the VLM baseline. These findings highlight the practical advantage of separating language reasoning from object detection, thereby avoiding the computational burden of end-to-end multimodal inference.

Future work will evaluate open-vocabulary object detectors, such as YOLO-World [20] and OWL-ViT [21], to enable text-conditioned detection beyond the predefined object classes used in YOLO-SC. In particular, TensorRT-optimized OWL-ViT deployments such as NanoOWL [22] may be investigated on Jetson platforms. This extension should be evaluated by jointly measuring open-vocabulary detection accuracy, end-to-end latency, and resource usage on the target embedded hardware.

**Acknowledgements.** The authors acknowledge the support provided by the Directorate of Research, Universitas Gadjah Mada.

# Statements and Declarations

**Funding.** This work was supported by the Directorate of Research, Universitas Gadjah Mada, through the Program Peningkatan Academic Excellence 2025 (Scheme B).

**Competing interests.** The authors have no relevant financial or non-financial interests to disclose.

**Ethics approval and consent to participate.** Not applicable. This study did not involve human participants, personal data, animals, or biological materials.

**Consent for publication.** Not applicable.

**Data availability.** The data generated and analyzed during the current study are available from the corresponding author on reasonable request.

**Materials availability.** Not applicable.

**Code availability.** The implementation code is available from the corresponding author on reasonable request.

**Author contributions.** Ahmad Ataka contributed to the conception, implementation, evaluation, supervision, interpretation, and preparation of the manuscript. Dino Accoto, Igi Ardiyanto, and Jun Miura contributed to the conception, supervision, and evaluation. Yanayir Rifai and Yusuf Badriawan contributed to the implementation, evaluation, and interpretation. Burhanudin contributed to the implementation. All authors reviewed and approved the final manuscript.

**Use of artificial intelligence-assisted technologies.** AI-assisted tools were used only to support language editing and document formatting. They were not used to generate scientific results, data, analyses, or conclusions. All content was reviewed and verified by the authors, who take full responsibility for the manuscript.

**Author biographies.** **Ahmad Ataka** is with the Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia. His research interests include robotics, control systems, autonomous navigation, and human–robot interaction.

**Yanayir Rifai** is with the Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia. His research interests include service robotics, visual perception, and autonomous navigation.

**Yusuf Badriawan** is with the Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia. His research interests include service robotics, robot perception, and intelligent robotic systems.

**Burhanudin** is with the Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia. His research interests include robotic-system implementation and software integration.

**Igi Ardiyanto** is with the Department of Electrical Engineering and Information Technology, Universitas Gadjah Mada, Yogyakarta, Indonesia. His research interests include robotics, artificial intelligence, perception, and autonomous systems.

**Jun Miura** is with the Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Japan. His research interests include intelligent robotics, computer vision, and autonomous robotic systems.

**Dino Accoto** is affiliated with Flanders-Make@KULeuven and the Robotics, Automation and Mechatronics group, Department of Mechanical Engineering, KU Leuven, Belgium. His research interests include robotics, mechatronics, and human-centered robotic systems.

# References


[1] Garcia-Haro, J.M., Oña, E.D., Hernandez-Vicen, J., Martinez, S., Balaguer, C.: Service robots in catering applications: A review and future challenges. Electronics **10**(1), 47 (2021) https://doi.org/10.3390/electronics10010047

[2] Holland, J., Kingston, L., McCarthy, C., Armstrong, E., O'Dwyer, P., Merz, F., McConnell, M.: Service robots in the healthcare sector. Robotics **10**(1), 47 (2021) https://doi.org/10.3390/robotics10010047

[3] Vasconez, J.P., Kantor, G.A., Auat Cheein, F.A.: Human–robot interaction in agriculture: A survey and current challenges. Biosystems Engineering **179**, 35–48 (2019) https://doi.org/10.1016/j.biosystemseng.2018.12.005

[4] Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press, Cambridge, MA (2005)

[5] Sapkota, R., Cheppally, R.H., Sharda, A., Karkee, M.: YOLO26: Key Architectural Enhancements and Performance Benchmarking for Real-Time Object Detection (2025). https://doi.org/10.48550/arXiv.2509.25164

[6] Kim, C.Y., Lee, C.P., Mutlu, B.: Understanding large-language model (LLM)-powered human–robot interaction. In: Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction, pp. 371–380. ACM, New York, NY, USA (2024). https://doi.org/10.1145/3610977.3634966

[7] Stevenson, S., Shek, S.H., Bajit, E., Nath, D., Vela, P.A.: LLM-enhanced mobile robot navigation and scene description for indoor environments. In: 2025 IEEE Opportunity Research Scholars Symposium (ORSS), pp. 1–5. IEEE, Piscataway, NJ, USA (2025). https://doi.org/10.1109/ORSS66051.2025.11121634

[8] Zu, W., Song, W., Chen, R., Guo, Z., Sun, F., Tian, Z., Pan, W., Wang, J.: Language and sketching: An LLM-driven interactive multimodal multitask robot navigation framework. In: 2024 IEEE International Conference on Robotics and Automation (ICRA), pp. 1019–1025. IEEE, Piscataway, NJ, USA (2024). https://doi.org/10.1109/ICRA57147.2024.10611462

[9] Dorbala, V.S., Mullen, J.F., Manocha, D.: Can an embodied agent find your "cat-shaped mug"? LLM-based zero-shot object navigation. IEEE Robotics and Automation Letters **9**(5), 4083–4090 (2024) https://doi.org/10.1109/LRA.2023.3346800

[10] Sutra, N., Ataka, A., Cahyadi, A.I.: Quadcopter trajectory generation based on large language model. In: 2025 IEEE International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), pp. 1–6. IEEE, Piscataway, NJ, USA (2025). https://doi.org/10.1109/AIMS66189.2025.11229484

[11] Bintoro, K., Rizqi, A.A.A., Wahyunggoro, O.: Soft robotic palm control based on large language model. In: 2025 IEEE International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), pp. 1–6. IEEE, Piscataway, NJ, USA (2025). https://doi.org/10.1109/AIMS66189.2025.11229504

[12] Shah, D., Osiński, B., Ichter, B., Levine, S.: LM-nav: Robotic navigation with large pre-trained models of language, vision, and action. In: Proceedings of the 6th Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 205, pp. 492–504. PMLR, Brookline, MA, USA (2023). https://proceedings.mlr.press/v205/shah23b.html

[13] Liu, J.X., Shah, A., Konidaris, G., Tellex, S., Paulius, D.: Lang2LTL-2: Grounding spatiotemporal navigation commands using large language and vision-language models. In: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2325–2332. IEEE, Piscataway, NJ, USA (2024). https://doi.org/10.1109/IROS58592.2024.10802696

[14] Zhao, X., Lyu, X., Li, X.: AgriVLN: Vision-and-Language Navigation for Agricultural Robots (2025). https://doi.org/10.48550/arXiv.2508.07406

[15] Zhou, G., Hong, Y., Wang, Z., Wang, X.E., Wu, Q.: NavGPT-2: Unleashing navigational reasoning capability for large vision-language models. In: Computer Vision – ECCV 2024, pp. 260–278. Springer, Cham, Switzerland (2024). https://doi.org/10.1007/978-3-031-72667-5_15

[16] Kravtsov, A.N., Goryachkin, B.S.: Integration of large language models for autonomous navigation of a mobile robot. In: 2025 7th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE), pp. 1–6. IEEE, Piscataway, NJ, USA (2025). https://doi.org/10.1109/REEPE63962.2025.10971154

[17] Liu, X., Farid, A., Amano, T., Rizk, H., Yamaguchi, H.: LLM-powered embodied intelligence for socially-aware robot navigation in human–robot interaction. In: Proceedings of the 19th International Symposium on Spatial and Temporal Data, pp. 145–148. ACM, New York, NY, USA (2025). https://doi.org/10.1145/3748777.3748807

[18] Liu, H., Zhu, Y., Kato, K., Kondo, I., Aoyama, T., Hasegawa, Y.: LLM-Based Human–Robot Collaboration Framework for Manipulation Tasks (2023). https://doi.org/

10.48550/arXiv.2308.14972

[19] Ataka, A., Rifai, Y.: Spatial context-aware VLM-assisted goal reasoning for navigation in guide service robots. In: 2026 IEEE International Conference on Human-Machine Systems (ICHMS), pp. 446–451. IEEE, Piscataway, NJ, USA (2026). https://doi.org/10.1109/ICHMS69701.2026.11602162

[20] Cheng, T., Song, L., Ge, Y., Liu, W., Wang, X., Shan, Y.: YOLO-world: Real-time open-vocabulary object detection. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16901–16911. IEEE, Piscataway, NJ, USA (2024). https://doi.org/10.1109/CVPR52733.2024.01599

[21] Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., Wang, X., Zhai, X., Kipf, T., Houlsby, N.: Simple open-vocabulary object detection with vision transformers. In: Computer Vision – ECCV 2022, pp. 728–755. Springer, Cham, Switzerland (2022). https://doi.org/10.1007/978-3-031-20080-9_42

[22] NVIDIA: NanoOWL: TensorRT-Optimized Open-Vocabulary Object Detection on Jetson Orin. GitHub repository (2026). https://github.com/NVIDIA-AI-IOT/nanoowl Accessed 14 September 2026