Multi-Task Visual Perception Network with LLM Conditioning for Autonomous Navigation

Praveen Kumar, K.R. Guruprasad†, and Tushar Sandhan
Department of Electrical Engineering, †Department of Mechanical Engineering
Indian Institute of Technology Kanpur, India
{praveenk20, krgprao, sandhan}@iitk.ac.in

***Abstract*—Long-term navigation for service robots faces critical challenges like the accumulation of odometry drift and sensor error, which progressively degrade 2D maps and renders traditional path planning algorithms (e.g., A*, RRT*, DiPPer, ViT-A*) ineffective over time. To address this, we propose a user-friendly, interactive framework that eliminates the reliance on globally consistent maps. Our approach integrates visual perception with Large Language Models (LLM) to interpret user commands via text or voice. Instead of relying on a drift-prone global map, the system generates a sequential action plan based on local visual cues and egocentric geometric instructions. These action plans are executed sequentially, allowing the robot to navigate known and unknown environments safely. By resetting localization relative to immediate targets, our framework effectively works with a minimum accumulation drift strategy, ensuring accurate, efficient, and collision-free navigation without the maintenance overhead of traditional mapping. Experiments on real-world and simulated data have shown significant improvements over other methods. Our source code is publicly accessible at https://github.com/PraveenSingh24/VL-Navigation.**

***Index Terms*—Service Robots, Vision-Language Navigation, LLM, Egocentric Navigation, Odometry Drift mitigation.**

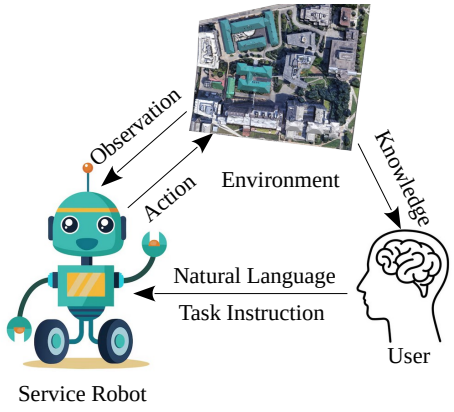


Fig. 1. Overview of the proposed scenario. The user leverages prior spatial knowledge of the environment to provide natural language task instructions to the service robot. The robot interprets these instructions and interacts with the environment through continuous observation and action, executing navigation tasks sequentially without relying on a global map.

I. Introduction

Service robots are increasingly deployed in dynamic indoor settings, such as residences and offices, where they must operate autonomously for extended periods. However, these environments are characterized by narrow passages and complex visual clutter, which often degrade traditional map-based navigation systems.

We propose a paradigm shift towards human-centric navigation, inspired by how humans verbally coordinate tasks. Consider an example scenario where a robot's admin commands a robot to "fetch a water bottle from the kitchen," or in a navigation context, a user might guide the robot using egocentric geometric and visual mark instructions such as: "Move 5 meters forward, take a right turn, and stop near the statue (Elephant)." Robots use natural language to identify local visual landmarks and geometric subgoals during a mission. It resets their navigation at each stage, significantly reducing long-term drift accumulation. Path planning has evolved from classical search-based algorithms (A* [1], D* Lite [2]) and sampling-based techniques (RRT [3], PRM [4]) to modern hybrid architectures (Informed RRT* [5], Hybrid A* [6]) and data-driven neural planners (ViT-A* [7], DiPPer [8]). While these modern approaches offer improved efficiency or adaptability, they remain constrained by a common bottleneck: the requirement for a static, globally accurate map. Reliance on globally consistent mapping constitutes a primary cause of failure for continuous service operations in an open, dynamic, and unstructured home environment [9]. Traditional SLAM frameworks assume a static world, which degrades rapidly in dynamic human-centric environments where furniture and layout change frequently [10]. Further, the pneumatic tire slip and skid-steering produce non-linear odometry errors that state-of-the-art SLAM algorithms cannot entirely suppress over long-duration deployments [11]. In visually complex, narrow passageways, the odometry drift can lead to collision or mission failure. Consequently, the global map differs from real-world settings, diminishing the ability to localize and reach goals and increasing the need for mapless, egocentric navigation strategies [12]. Our work does not use a static (Known) and an online-explored (Unknown) global map for path finding, as in traditional methods and latest learning models do; instead, it uses them to demonstrate our system's performance. Additionally, these systems demand precise metric goal coordinates (x, y), creating a usability gap; human users naturally communicate tasks via semantic instructions (e.g., "go to the kitchen") rather than providing global coordinates on a distorted digital map. As discussed in the recent survey on tasks and methods in vision-and-

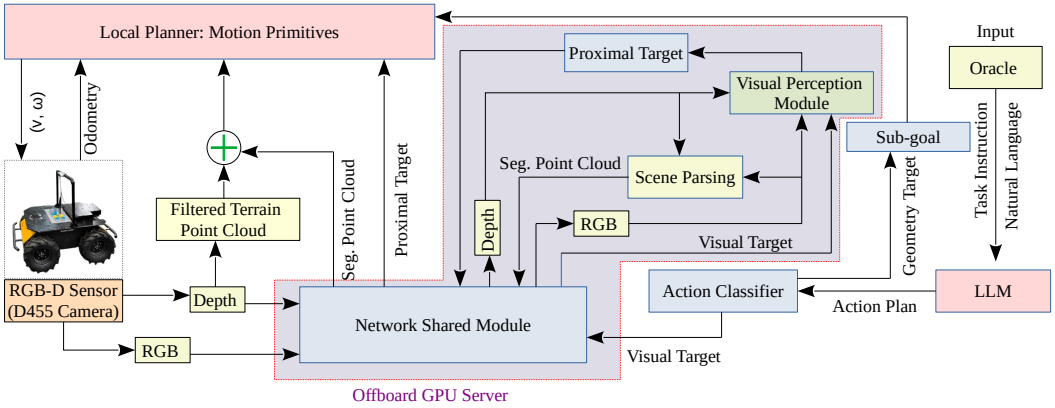


Fig. 2. Overview of the proposed vision-language navigation framework, which is distributed across two nodes: the onboard client (Husky A200) and an offboard GPU server for heavy computation. The workflow begins with Oracle Task Instructions (Natural Language), which are processed by LLM to generate an action plan. The Action Classifier parses this plan into geometric or visual sub-goals. To enable safe navigation, the Scene Parsing module generates a Segmented (Seg.) Point Cloud—representing specific semantic regions—which is concatenated with the filtered terrain point cloud. Finally, the Motion Primitive Local Planner computes optimal velocity commands (v, ω) to drive the robot toward the Proximal Target.

language navigation [13], the field is shifting towards agents that can seamlessly bridge the gap between visual perception and natural language communication. Recent works such as DyNaVLM [14] and VLingNav [15] investigate vision-language navigation using learned memory and reasoning mechanisms, with VLingNav further introducing an adaptive chain-of-thought (AdaCoT) strategy for dynamic decision-making. AdaVLN [16], in contrast, provides a simulator and benchmark for evaluating such approaches. At the same time, these approaches are effective in controlled environments. These methods often rely on computationally intensive representations or extensive training, limiting their robustness and deployability in real-world settings. We align with this direction by introducing a framework in which the robot acts as an intelligent assistant, processing natural-language user commands and combining current and past visual context.

Rather than planning a long path on a noisy and drifted map, we propose the LLM decomposes the user's intent into a sequence of action tasks. It enables the service robot to navigate to a semantic goal (e.g., a specific object or room) accurately and efficiently by grounding navigation in local visual observations rather than global coordinates. Our framework can be deployed to provide continuous uninterrupted services in a critical, high-risk mission. Even in a narrow passage setup, our framework worked well.

II. Proposed Method

This section details the proposed vision-geometric navigation framework designed for wheeled service robots operating in human-centric environments. To ensure accessibility for non-expert users in daily scenarios, the system interacts via natural language, allowing operators to issue intuitive task instructions without requiring technical expertise. The framework intelligently interprets these guidelines to navigate workspaces containing critical visual constraints, effectively filling the gap between human intent and robotic execution. As shown in Fig. 2, the architecture integrates an LLM (Gemini Pro), a Visual Perception Module, and a Motion Primitive Local Planner to execute these instructions while mitigating odometry drift in constrained spaces.

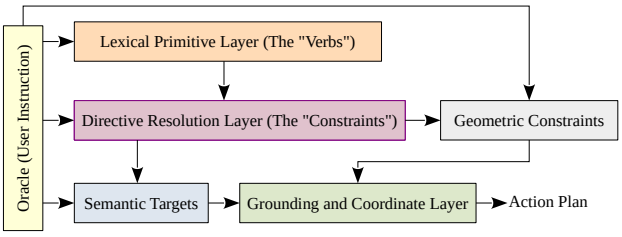


Fig. 3. The multi-layered architecture for converting user task instructions into robot action plan. The system resolves lexical ambiguity and distinguishes between geometric goal and semantic target constraints before grounding them into a unified robot system for execution.

A. Task Instruction Processing Module

For service robots to evolve from industrial tools to ubiquitous assistants in homes and offices, they must be accessible to users unfamiliar with technical robotics. For instance, the system parses:

- Object Retrieval: "Bring a water bottle from the kitchen."
- Hybrid Geometric & Semantic Navigation: "Go ahead 10m, turn right at the Lotus statue, and proceed to the pharmacy."
- Social & Contextual Interaction: "Hi Buddy, please deliver this file to Mr. David and report back to me."

We propose a novel Hierarchical Task Network (HTN) architecture as illustrated in Fig. 3, designed for translating natural language instructions into robot actions in service and industrial robots. The process begins at the Lexical Primitive (e.g., FIND, MOVE) Layer (Level-1), where core action verbs are extracted to identify task intent. This is

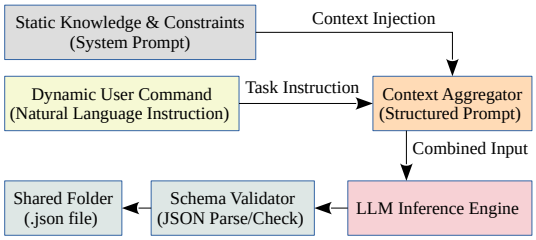


Fig. 4. Structured prompt is created via injecting static system constraints into the context window alongside the user task instruction. LLM translates the natural language request into a machine-readable format.

followed by the Directive Resolution Layer (Level-2), which bifurcates the instruction into dual pathways: an egocentric branch for geometric constraints relative to the robot's local frame, and an allocentric branch for semantic targets within the global coordinate frame. Finally, the Grounding and Coordinate Layer (Level-3) parameterizes these directives into precise spatial data and dispatches them to specialized ROS execution nodes via a unified, serialized action plan. To ensure reliability and safety, we use Gemini Pro for its low-latency multimodal reasoning; hallucinations and undefined behaviors are mitigated by constraining outputs to a closed action set and enforcing a schema validator (Fig. 4). The architecture illustrated in Fig. 4 adopts prompt engineering via our action graphs to bridge the gap between unstructured natural-language commands and deterministic robotic control. A composite prompt is constructed by combining the user's instruction with a set of domain-specific constraints, including available object categories and predefined action primitives.

The resulting output is therefore not free-form text but a deterministic symbolic representation that the robot's control pipeline can directly interpret. The Action Classifier is implemented as a rule-based semantic parser that transforms natural language task instructions into an ordered sequence of executable commands. When a visual subgoal cannot be found, the robot initiates a bounded exploration within a predefined obstacle-free region to re-detect the next goal. For an unsuccessful attempt, the system queries the user with contextual feedback and regenerates the action plan accordingly.

B. Action Plan Decoding and Subgoal Estimation

The action plan encodes high-level navigation commands in a structured format suitable for robotic execution. We classify these commands into two categories: perceptual targets, which require visual reasoning, and egocentric geometric targets, which define motion primitives relative to the robot's current pose.

Egocentric geometric targets are computed directly from the instruction semantics through context parsing and are translated into spatial subgoals expressed in the robot's egocentric coordinate frame. These subgoals guide the navigation controller toward the desired waypoint in a step-wise manner. Let the robot's pose in the global coordinate frame be defined as $\mathbf{R}_r = [x_r, y_r, \theta_r]^T$, where (x_r, y_r) denotes the position and θ_r represents the heading. Correspondingly, the egocentric motion command extracted from the instruction is represented as $\Delta\mathbf{p} = [\Delta x, \Delta y]^T$, where Δx and Δy define the required displacement within the robot's local coordinate frame.

The global target position $\mathbf{G}_c = (x_g, y_g)$ is computed by transforming the egocentric motion into the global frame using the relation $\mathbf{G}_c = [x_r, y_r]^\top + \mathbf{R}(\theta_r)[\Delta x, \Delta y]^\top$, where the rotation matrix $\mathbf{R}(\theta_r) = \begin{bmatrix} \cos\theta_r & -\sin\theta_r \\ \sin\theta_r & \cos\theta_r \end{bmatrix}$ aligns the local displacement with the robot's current heading θ_r relative to the global origin. This computed global coordinate $\mathbf{G}_c$ is then published as a waypoint to a modified version of the local planner described in [17], which we adapt to our navigation setup and task objectives. This allows the robot to execute the relative motion specified by the natural language instruction.

Visual perception targets are categorized into two fundamental perception tasks: (i) object detection and (ii) person re-identification (ReID). The selection of the appropriate perception module is automatically determined based on the action keyword extracted from the generated action plan. For object-centric goals, the framework employs a YOLOv8-based detector, while person-specific tasks are handled using an OSNet-based ReID model [18] trained on our custom dataset. During execution, RGB images acquired from the Intel RealSense D455 camera are processed online by the selected perception network to localize the target and generate a bounding box, which is subsequently used for downstream spatial reasoning and navigation. Let the bounding box be defined as $\mathcal{B} = (u_{\min}, v_{\min}, u_{\max}, v_{\max})$, and its center pixel be (u_c, v_c). A robust depth estimate is obtained by computing the median depth value within the bounding box region, filtered to remove invalid or noisy measurements

$$Z = \text{median}\,\{D(u,v) \mid (u,v) \in \mathcal{B},\ Z_{\min} < D(u,v) < Z_{\max}\}\,. \tag{1}$$

Using the camera intrinsic matrix $\mathbf{K}$, the 3D coordinates of the detected object in the camera optical frame are recovered as $X_c = (u_c - c_x)Z/f_x$, $Y_c = (v_c - c_y)Z/f_y$, and $Z_c = Z$. Following the standard ROS coordinate convention, these coordinates are transformed into the robot base frame as $X_r = Z_c$, $Y_r = -X_c$, and $Z_r = 0$. The resulting point is then transformed into the global coordinate frame using the robot's pose and the corresponding rigid-body transformation $[x_m, y_m]^\top = \mathbf{R}(\theta_r)[X_r, Y_r]^\top + \mathbf{t}$, where $\mathbf{R}(\theta_r)$ and $\mathbf{t} = [x_r, y_r]^T$ represent the rotation and translation from the robot frame to the map frame. To ensure safe navigation, the robot does not directly approach the detected object but instead computes a subgoal at a fixed stopping distance d_s from the target. Let (x_r, y_r) denote the robot position and (x_m, y_m) the target position in the map frame. The final navigation goal is computed by first determining the Euclidean distance $d = \sqrt{(x_m - x_r)^2 + (y_m - y_r)^2}$ and bearing $\theta = \tan^{-1}((y_m - y_r)/(x_m - x_r))$ to the target. Subsequently, the goal coordinates, adjusted for a safe stopping distance d_s, are derived as $x_g = x_r + (d - d_s)\cos\theta$ and $y_g = y_r + (d - d_s)\sin\theta$. This target (x_g, y_g) is then published as a navigation waypoint, allowing the robot to approach the detected object while

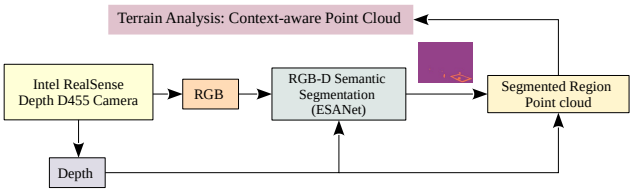


Fig. 5. Visual perception to point cloud generation pipeline. The RGB–D stream acquired from the Intel RealSense D455 camera is processed by an RGB-D semantic segmentation network (ESANet) to extract task-relevant regions. The resulting segmented mask is projected into 3D space using depth information to generate a semantic point cloud, which is subsequently utilized for terrain analysis and context-aware navigation.

maintaining a safe and controlled distance.

C. Context-Aware Environmental Perception

The robot perceives and interprets its surroundings to support safe, task-aware navigation. The perception module enables the robot to identify semantically meaningful elements such as color-marked restricted zones, electrical cables lying on the ground, objects placed along the navigation path (e.g., mobile phones or documents), and chemically contaminated or hazardous surface regions. The relevance of these elements may vary depending on the operational context, environmental conditions, and user-defined task priorities.

D. Motion Primitive Generation and Selection

To ensure safe and efficient autonomous navigation in complex environments, the local planner evaluates a discrete set of candidate trajectories against semantic perception data and geometric constraints. This process is divided into two stages: the retrieval and transformation of pre-computed motion primitives, and the optimal selection based on a context-aware cost function.

1) Pre-computed Path Library and Transformation: Instead of solving the boundary value problem (BVP) for real-time trajectory generation, which can be computationally expensive, we use a library of precomputed motion primitives. As shown in Fig. 6 (a), this library consists of 343 kinematically feasible trajectories generated offline. These primitives represent a diverse set of maneuvers that respect the mobile robot's non-holonomic constraints.

During runtime, the planner adapts these static templates to the robot's dynamic state $\mathbf{x}_t = [x, y, \theta, v]^T$. The selected primitives are transformed from the library frame to the robot's local coordinate frame using a rigid-body transformation and scaled according to the current vehicle velocity v_{curr}. It ensures that the search space expands faster to look further ahead and contracts more slowly for precise maneuvering.

2) Feasibility and Collision Checking: The transformed candidates are evaluated for feasibility using a temporary local representation—an egocentric voxel grid constructed from real-time depth frame data and ESANet [19]. Unlike persistent global SLAM maps used for long-term planning, this transient representation is employed solely for immediate collision-free trajectory selection. A lookup table maps each voxel index to the subset of motion primitives that intersect it. If a primitive traverses an occupied voxel—defined as a point where the terrain height or semantic class indicates an untraversable obstacle—it is flagged as invalid and removed from the candidate set.

3) Optimal Path Selection: The optimal trajectory τ^* is selected from the set of candidate motion primitives by minimizing a composite cost function $J(\tau)$. The scoring mechanism balances goal alignment with safety from both physical and semantic hazards,

$$J(\tau) = w_g \cdot J_{goal}(\tau) + w_o \cdot J_{obs}(\tau) + w_s \cdot J_{semantic}(\tau) \quad (2)$$

where $J_{goal}(\tau)$ represents the heading alignment error, quantifying the deviation between the robot's orientation at the end of the primitive $(x_{end}, y_{end}, \theta_{end})$ and the bearing to the subgoal $\mathbf{G}_c$ and $\text{atan2}(\cdot,\cdot)$ denotes the four-quadrant inverse tangent operator, and $\text{wrap}(\cdot)$ normalizes the angular difference to $[-\pi, \pi]$.

$$J_{goal}(\tau) = |\text{atan2}(y_g - y_{end}, x_g - x_{end}) - \theta_{end}| \quad (3)$$

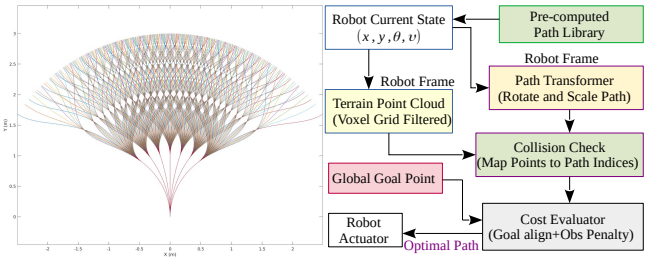
and $J_{obs}(\tau)$ represents the physical obstacle penalty. It is derived from the minimum Euclidean distance d_{phy}, between the robot's position $\mathbf{p}(t) = [x(t), y(t)]^T$ along the trajectory and the filtered point cloud generated from the RGB-D camera's depth frame. To ensure collision-free navigation, the cost scales exponentially with proximity,

$$J_{obs}(\tau) = \exp\left(-\alpha \cdot \min_{t\in\tau} d_{phy}(\mathbf{p}(t))\right) \quad (4)$$

and $J_{semantic}(\tau)$ acts as a virtual obstacle penalty. The visual perception module converts restricted zones (e.g., non-traversable semantic labels) into virtual point clouds with a height of 0.5m. The planner treats these exactly as physical barriers, applying an identical penalty structure based on the distance d_{sem} to the nearest virtual obstacle,

$$J_{semantic}(\tau) = \exp\left(-\beta \cdot \min_{t\in\tau} d_{sem}(\mathbf{p}(t))\right) \quad (5)$$

where w_g, w_o, and w_s are weighting constants, and $alpha$, β are decay constants. These $alpha$ and β are tuned to the repulsiveness of geometric and semantic barriers, respectively. All these parameters are fixed across experiments and empirically tuned. Finally, the candidate trajectory τ^* minimizing the aggregate cost is selected. The control inputs from the selected optimal trajectory, the linear (v) and angular (ω) velocities respectively, are extracted and published to the robot's low-level controller for imediate action on task instruction. Fig. 7 shows black arrows that denote the robot's traversed path between the robot and its next target. The intermediate targets are G_1, G_2, and G_3, while D denotes the user-specified final destination. The light-yellow circular regions shown in Fig. 7 represent safety zones centered around each target, whose radii are set by the user based on surrounding visual and geometric context. The odometry drift at each subgoal, denoted as E_{g1}, E_{g2}, E_{g3}, and E_d, is computed during navigation, with the



(a) (b)

Fig. 6. (a) Set of candidate motion trajectories produced using the motion primitive framework. These trajectories represent feasible forward motions under kinematic constraints and are evaluated against environmental and semantic constraints to select an optimal navigation path. (b) Data flow of the local planner. The process begins by synchronizing the pre-computed path library and sensor data to the robot's current state. Feasible paths are filtered through a voxel-grid collision check, and the final trajectory is chosen by minimizing a cost function to ensure safe and efficient navigation.

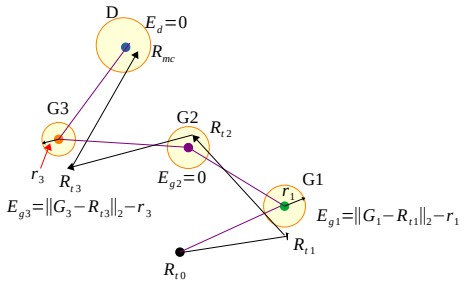


Fig. 7. Illustration of odometry drift estimation during navigation. The robot moves through intermediate subgoals G_1–G_3 toward the final destination D, and the drift is computed based on the deviation between the robot pose and the corresponding target.

final drift evaluated at the destination point D. In our work, the navigation is performed without maintaining a global map, localization relies solely on wheel encoder and IMU measurements, and no loop closure is applied.

III. EXPERIMENTS

To evaluate the robustness and real-time performance of our framework, validation was conducted in both a high-fidelity simulation environment and a real-world experimental setup. Consistent with the distributed architecture described previously, both setups use a Client-Server architecture to balance computational load efficiently.

In the Simulation setup, the Client node serves as the robot controller and runs on a desktop workstation equipped with an Intel Core i7-4770 processor. The Server node, responsible for heavy perception tasks, runs on a separate high-performance workstation equipped with an NVIDIA RTX A4000 GPU.

Simulation Results and Analysis: The proposed framework successfully guided the differential-wheeled robot to its final destination (4. Bicycle) within the Gazebo simulation environment, demonstrating robust autonomous navigation, following the complex multi-step instructions. The robot sequentially localized and approached each semantic landmark.

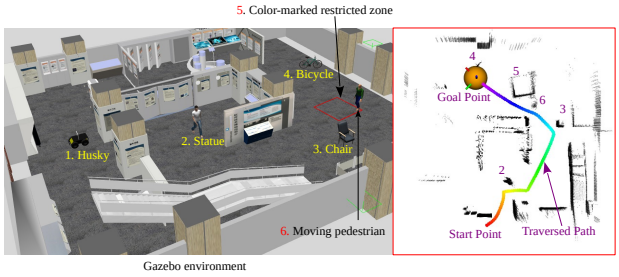


Fig. 8. Experimental validation in the Gazebo simulation environment. (Left) The 3D test workspace containing the Husky A200 robot and various semantic obstacles labeled as: 2) Statue, 3) Chair, 4) Bicycle, 5) Color-marked restricted zone, and 6) Moving pedestrian. (Right) Top-down view of the generated point cloud map illustrating the autonomous navigation performance, showing the Start Point, Goal Point, and the resulting collision-free Traversed Path

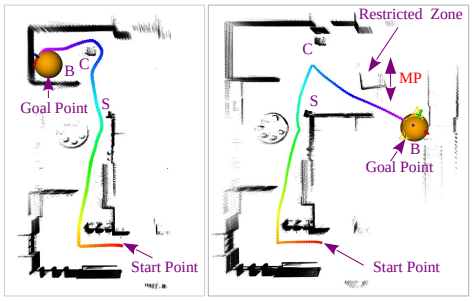


Fig. 9. Explored environment and traversed paths for two distinct navigation tasks. The left and right panels display the top-down occupancy map, where black regions represent physical barriers and context-aware constraints. The robot successfully navigates from the Start Point to the Goal Point by executing a sequence of semantic actions. Key landmarks are labeled as S (Statue/Human), C (Visitor Chair), and B (Bicycle). The planner also accounts for dynamic and static constraints, such as the Moving Pedestrian (MP) and the designated Restricted Zone.

As illustrated in Fig. 8, the traversed path is optimal and collision-free. Crucially, the system demonstrated advanced context-aware perception: it strictly avoided the color-marked restricted zone, treating it as a critical safety constraint, and dynamically navigated around a moving pedestrian without interrupting the mission. These experimental results validate the framework's ability to interpret natural language commands while prioritizing environmental safety and efficiency.

Odometry data is reset after each successful mission completion, limiting drift to the distance between the start point and its target. The maximum subgoal distance is 10 m due to depth data quality. Geometric targets may introduce additional drift for navigation in a featureless Corridor. RGB-D data do not constrain them. Heavy computational SLAM is not required for short-range navigation, but it can be integrated for accurate, precision-critical tasks.The proposed framework does not rely on a persistent global SLAM map. It is a mapless navigation algorithm that enables a wheeled robot to operate solely from real-time camera data and local visual cues. A temporary map frame is introduced for qualitative observation and is fully decoupled from the navigation process. It serves as a fixed reference to depict the vehicle's trajectory in real time,

TABLE I
COMPARISON OF MODERN NAVIGATION PARADIGMS (EXTENDED). ✓ SUPPORTED, ✗ NOT SUPPORTED, △ PARTIAL.

Method	Global Map	Drift Sensitivity	Long-Term Nav.	Unknown Env.	Semantic Understanding	Context Awareness	User Interaction	Replanning
A* / RRT*	✓	✓	✗	✗	✗	✗	✗	Limited
DiPPer	✓	✓	△	✗	✗	✗	✗	Limited
ViT-A*	✓	✓	△	△	△	✗	✗	Limited
Vision-Aided A* [20]	✓	✓	✗	✓	✗	✓	✗	Online
SemExp (Semantic Exploration) [21]	✗	△	△	✓	✓	✓	✗	Moderate
CARe (Context-Aware Replanning) [22]	✗	△	✓	✓	✓	✓	✗	Advanced
SEMNAV (Semantic Segmentation) [23]	✗	✗	✓	✓	✓	✓	✗	Advanced
Multi-Agent Semantic Nav [24]	✗	✗	✓	✓	✓	✓	✗	Advanced
Ours	✗	✗	✓	✓	✓	✓	✓	**Online**

TABLE II
QUALITATIVE EVALUATION OF LLM-GUIDED INSTRUCTION PARSING.

Case 1: Complex Navigation
User Input: *"Buddy go to the statue (human), take a right turn and move a little ahead, then find the visitor chair to reach close to it, and on the right you may see a bicycle."*

LLM Output:
`[{"action": "FIND", "target": "statue (human)", "id": 0},`
`{"action": "MOVE", "desc": "turn right...", "f_dist": 2.5, "yaw": -1.57},`
`{"action": "FIND", "target": "visitor chair", "id": 56},`
`{"action": "FIND", "target": "bicycle", "id": 1}]`
Status: Success

Case 2: Metric Constraints
User Input: *"Buddy take left turn, move 4 m ahead, then find the standing person to reach close to it, In your right you may see the visitor chair follow it. Finally go at spot where bicycle is standing."*

LLM Output:
`[{"action": "MOVE", "desc": "left turn 4m", "f_dist": 4.0, "yaw": 1.57},`
`{"action": "FIND", "target": "standing person", "id": 0},`
`{"action": "FIND", "target": "visitor chair", "id": 56},`
`{"action": "FIND", "target": "bicycle", "id": 1}]`
Status: Success

without contributing any spatial information for path planning or goal execution.

Real-World Experimental Results and Analysis: The real-world experiments are conducted on the Husky robot, which is powered by an onboard NVIDIA Jetson AGX embedded module. The server computing device (optional) is used when the robot system is insufficient for the real-time execution of the semantic visual task.

This section evaluates the system's performance in a visually rich, constrained real-world environment ($10\,\mathrm{m} \times 5.5\,\mathrm{m}$), as illustrated in Fig. 10. The environment is cluttered with common physical barriers and objects, including water bottles, a red-marked box, cables, chairs, and potted plants. This indoor setting serves as a rigorous testbed for multiple experimental trials and system validation.

Fig. 12 (a) demonstrates the robot's ability to navigate context-aware settings. The robot successfully identifies and avoids the red-marked box (prohibited zone), then moves toward the visual target (a chair) and finally halts at a definite distance threshold. The top-down point cloud view in Fig. 12 (a) reveals the detection of smaller objects such as a Book, a Cable, and a 250ml bottle. Notably, our proposed

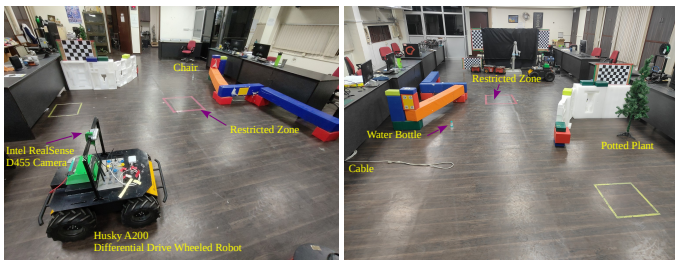


Fig. 10. Indoor experimental environment with the Husky A200 robot and onboard Intel RealSense D455 camera. The scene includes multiple visual targets and restricted regions in a confined workspace used for real-world navigation evaluation.

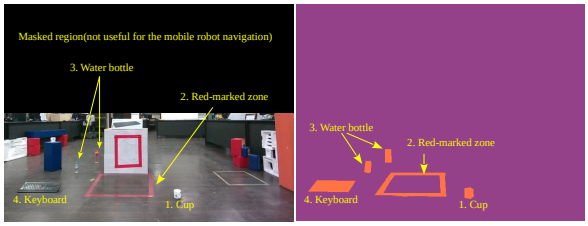


Fig. 11. ESANet model input and output: The frame (left) shows the settings: the marked objects are placed on the surface, the red-marked box is drawn on the ground, and the remaining items are at a significant height. The segmentation model output is shown (right), where we show the segmented region of the labeled items, which the user defines.

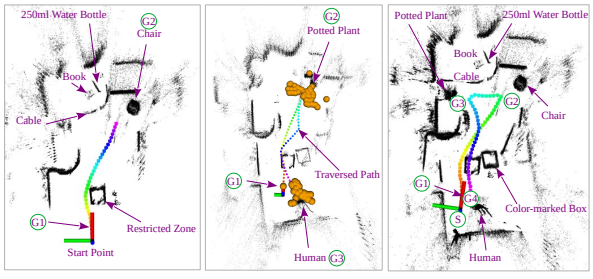
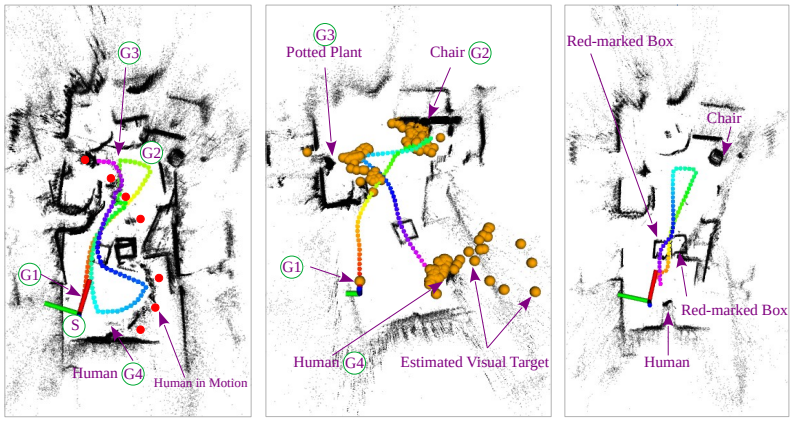


Fig. 12. Real-world navigation results across multiple trials. The robot starts from S and first moves toward an egocentric geometric goal (G1), followed by visually detected semantic targets (G2–G4). White regions indicate free space, black regions represent obstacles, and the colored path shows the executed trajectory.



(a) (b) (c)

Fig. 13. Experimental results of the visual navigation system. (a) The wheeled robot successfully tracking and following a dynamic human target. (b) Estimation of the fixed visual target point; note that despite the spatial spread of the estimated points, the robot successfully navigated to the visual destination. (c) The view of the red-marked box target, showing lateral shifts caused by the robot's to-and-fro motion during navigation.

TABLE III
INFERENCE TIME PERFORMANCE ACROSS COMPUTING PLATFORMS

Module	Task	Latency (ms)	Execution Mode
Server Platform (NVIDIA RTX A4000)			
RedNet	RGB-D Segmentation	29.2 ± 1.8	Continuous
ESANet	RGB-D Segmentation	34.8 ± 0.7	Continuous
OSNet	Person Re-ID	50 ± 5	On-demand
YOLOv8	Object Detection	12.3 ± 1.4	On-demand
LLM (Gemini Pro)	Instruction Parsing	150 ± 30	Per command
Onboard Platform (Husky A200 – Jetson AGX Xavier)			
RedNet	RGB-D Segmentation	87 ± 7	Continuous
ESANet	RGB-D Segmentation	108 ± 14	Continuous
OSNet	Person Re-ID	80 ± 10	On-demand
YOLOv8	Object Detection	38 ± 8	On-demand
LLM (Gemini Pro)	Instruction Parsing	190 ± 35	Per command

method effectively filters semantic noise located on the ground surface near the target chair, ensuring robust target identification.

The efficacy of the proposed approach in sequential navigation is demonstrated in Fig. 12 (b). The robot takes a user task-instruction to navigate and reach the egocentric geometric subgoals, then the semantic subgoals (G_2 and G_3). During this trajectory, the system detects a double red-marked box and visualizes multiple golden spheres, which represent estimated visual targets generated during navigation prior to reaching the current goal. It was observed that semantic segmentation noise, caused by varying lighting conditions and reflective surfaces, occasionally introduced false positive points into the environmental point cloud; however, the navigation system remained robust to these disturbances.

Finally, Fig. 12 (c) presents the results of a complex navigation task involving four distinct local goals. A minor map layout shift is observed in the global map; despite this slight deviation, the robot completed the multi-goal mission, confirming the system's robustness in complex, multi-goal navigation tasks.

TABLE IV
PERFORMANCE COMPARISON OF NAVIGATION MODALITIES UNDER DIFFERENT ENVIRONMENTAL CONSTRAINTS

Scenario	Constraint	Geometric	Visual	Hybrid (Ours)
Featureless Corridor	No visual landmarks	92.0	15.0	**95.0**
Symmetric Room	Geometric ambiguity	40.0	94.0	**96.0**
Occluded Target	Target not visible	85.0	30.0	**91.0**
Precise Positioning	Exact stopping required	96.0	65.0	**98.0**
Complex Mission	Sequential task execution	10.0	25.0	**88.0**

The proposed framework is validated in a dynamic scenario where the robot follows a moving human target, as depicted in Fig. 13 (a). Throughout this task, the planner successfully generates trajectories that avoid collisions and strictly adhere to user-defined environmental constraints, specifically avoiding the red-marked box and cables.

Fig. 13 (b) highlights challenges in narrow passages, where the path overlaps with the prohibited zone due to non-holonomic constraints and limited turning radius, further affected by the robot's large footprint (988 mm $\times$ 670 mm) and skid-steer kinematics; however, the framework remains platform-agnostic and can be readily deployed on smaller mobile robots. The figure also shows estimated target points (golden spheres), and despite their high variance, the algorithm effectively filters them to produce a safe, converging path.

The impact of sensor noise is analyzed in Fig. 13 (c). Here, artifacts arising from semantic segmentation-to-point cloud projection introduced geometric errors. Consequently, the robot momentarily grazed the red-marked zone during the initial motion from the start pose to the visual target (chair). However, the tracking from the potted plant to the human showed that the robot corrected and safely avoided the no-go area.

The average odometry drift error reported in Table V is computed as

$$\bar{E}_{\text{drift}} = \frac{1}{N} \sum_{t=1}^{N} \max \left(0,\ \left\| \mathbf{G}_d^{(t)} - \mathbf{R}_d^{(t)} \right\|_2 - d_s \right), \qquad (6)$$

TABLE V

PERFORMANCE EVALUATION RESULTS OVER 50 TRIALS PER SCENARIO. THE SIMULATION SCENARIOS (INDOOR, GARAGE, CAMPUS, FOREST, TUNNEL) ARE GIVEN IN DETAIL [17], [25]. THESE SETUPS ARE MODIFIED TO INCLUDE VISUAL OBJECTS AND SEMANTIC CONTEXT-AWARE ITEMS (SOFT COLLISION) FOR OUR SYSTEM PERFORMANCE EVALUATION. WE CONSIDER THE ENVIRONMENTAL STRUCTURE (HARD COLLISION) THAT OBSTRUCTS THE NAVIGATION PATH.

Environment	Success Rate (%)	Soft Collision (%)	Hard Collision (%)	Avg. Path Length (m)	Drift Error (m)
Simulation					
Indoor	100	10.0	0.0	12.4	0.03 ± 0.01
Garage	100	8.5	0.0	20.8	0.05 ± 0.03
Campus	98	10.0	0.0	30.5	0.05 ± 0.02
Forest	95	15.2	0.3	15.5	0.08 ± 0.01
Tunnel	98	14.0	0.0	25.2	0.05 ± 0.02
Real-World Experiments (Husky A200 Platform)					
Indoor Lab (Fig. 10)	95	15.0	0.5	12.4	0.1 ± 0.02
Outdoor Area	98	5.6	0.0	124.6	0.05 ± 0.01

where $\mathbf{R}_d^{(t)}$ denotes the robot's final position at the mission completion time for the t-th trial, and $\mathbf{G}_d^{(t)}$ represents the corresponding mission destination location in the global coordinate frame. The parameter d_s defines a safety radius centered at the goal location, within which the robot is considered to have successfully reached the target. This radius is selected based on the physical size of the object and task-specific safety requirements. The variable N denotes the total number of individual missions conducted under identical experimental settings.

IV. CONCLUSION

In this work, we present a reliable solution for human-language-driven navigation that leverages the synergy between LLM and semantic visual perception of an environment. It empowers service robots to interpret and perform natural language commands, making human-robot interaction more user-friendly and efficient. The experimental results demonstrate that the robot's performance in navigating with its admin intent in the dynamic real-world environment effectively distinguishes between physical barriers (e.g., Geometric objects that may obstruct the navigation path) and soft barriers (e.g., no-go zones and defined critical objects) using a context-aware framework. Our future task is to address the unstable environment, lighting conditions, and skidding surfaces handling, and plan to refine the candidate local trajectory generation to better accommodate the robot's kinematic constraints for non-jerking motion.

REFERENCES

[1] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[2] S. Koenig and M. Likhachev, "D* lite," in *Eighteenth national conference on Artificial intelligence*, pp. 476–483, 2002.

[3] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[4] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 2002.

[5] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *IEEE/RSJ international conference on intelligent robots and systems*, pp. 2997–3004, IEEE, 2014.

[6] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *ann arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[7] J. Liu, S. Lyu, D. Hadjivelichkov, V. Modugno, and D. Kanoulas, "Vit-a*: Legged robot path planning using vision transformer a," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, pp. 1–6, IEEE, 2023.

[8] J. Liu, M. Stamatopoulou, and D. Kanoulas, "Dipper: Diffusion-based 2d path planner applied on legged robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9264–9270, 2024.

[9] Y. Zhang, G. Tian, C. Zhang, C. Hua, W. Ding, and C. K. Ahn, "Environment modeling for service robots from a task execution perspective," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 10, pp. 1985–2001, 2025.

[10] P. Biber, T. Duckett, *et al.*, "Dynamic maps for long-term operation of mobile service robots," in *Robotics: science and systems*, pp. 17–24, 2005.

[11] T. Zhou, R. Fang, F. Wang, G. Wang, L. Wu, K. Su, L. Xu, H. Pu, and J. Luo, "Visual and lidar slam performance in campus scenes," in *2024, 3rd International Conference on Service Robotics (ICoSR)*, pp. 83–88, IEEE, 2024.

[12] F. Lin, Z. Ji, C. Wei, and H. Niu, "Reinforcement learning-based mapless navigation with fail-safe localisation," in *Annual Conference Towards Autonomous Robotic Systems*, pp. 100–111, Springer, 2021.

[13] J. Gu, E. Stefani, Q. Wu, J. Thomason, and X. Wang, "Vision-and-language navigation: A survey of tasks, methods, and future directions," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7606–7623, 2022.

[14] Z. Ji, H. Lin, and Y. Gao, "Dynavlm: Zero-shot vision-language navigation system with dynamic viewpoints and self-refining graph memory," *arXiv preprint arXiv:2506.15096*, 2025.

[15] S. Wang, Y. Luo, X. Chen, A. Luo, D. Li, C. Liu, S. Chen, Y. Zhang, and J. Yu, "Vlingnav: Embodied navigation with adaptive reasoning and visual-assisted linguistic memory," *arXiv preprint arXiv:2601.08665*, 2026.

[16] D. Loh, T. Bednarz, X. Xia, and F. Guan, "Adavln: Towards visual language navigation in continuous indoor environments with moving humans," *arXiv preprint arXiv:2411.18539*, 2024.

[17] C. Cao, H. Zhu, F. Yang, Y. Xia, H. Choset, J. Oh, and J. Zhang, "Autonomous exploration development environment and the planning algorithms," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 8921–8928, IEEE, 2022.

[18] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Omni-scale feature learning for person re-identification," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3702–3712, 2019.

[19] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H.-M. Gross, "Efficient rgb-d semantic segmentation for indoor scene analysis," in *2021 IEEE international conference on robotics and automation (ICRA)*, pp. 13525–13531, IEEE, 2021.

[20] P. Kumar and T. Sandhan, "Vision-aided online a* path planning for efficient and safe navigation of service robots," *arXiv preprint arXiv:2511.06801*, 2025.

[21] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4247–4258, 2020.

[22] P.-C. Ko, H.-T. Su, C. Chen, J.-F. Yeh, M. Sun, and W. H. Hsu, "Context-aware replanning with pre-explored semantic map for object navigation," in *8th Annual Conference on Robot Learning*, 2024.

[23] R. Flor-Rodríguez, C. Gutiérrez-Álvarez, F. J. Acevedo-Rodríguez, S. Lafuente-Arroyo, and R. J. López-Sastre, "Semnav: A semantic segmentation-driven approach to visual semantic navigation," *arXiv preprint arXiv:2506.01418*, 2025.

[24] X. Liu, D. Guo, H. Liu, and F. Sun, "Multi-agent embodied visual semantic navigation with scene prior knowledge," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3154–3161, 2022.

[25] F. Yang, C. Cao, H. Zhu, J. Oh, and J. Zhang, "Far planner: Fast, attemptable route planner using dynamic visibility update," in *2022 ieee/rsj international conference on intelligent robots and systems (iros)*, pp. 9–16, IEEE, 2022.