VISION-DRIVEN SOFT MANIPULATION
Integrating Soft Robotics and Computer Vision for Environment Interaction

Ramy Kachwaa
MBZUAI
Ramy.Kachwaa@mbzuai.ac.ae

Ibrahim Alsarraj
MBZUAI
Ibrahim.Alsarraj@mbzuai.ac.ae

Nakul Nibe
MBZUAI
Nakul.Nibe@mbzuai.ac.ae

Abstract

Robust object detection in dynamic and visually challenging environments remains a critical bottleneck. Conventional detectors limit their capacity to generalize to new objects or low-light conditions by depending on pre-defined datasets. We present an integrated robotic framework combining real-time RGB-D visual SLAM, autonomous navigation, soft robotic manipulation, and object detection enhanced by Vision-Language Models (VLMs), specifically GPT-4.1 Vision, to address these constraints. Using RTAB-Map SLAM and an RGB-D camera, our framework allows effective navigation with global and local planners of ROS Navigation Stack. Detectron2 performs initial object detection; then, GPT-4.1 Vision verifies detections, corrects misclassifications, and uses zero-shot semantic reasoning to find unseen objects. Particularly under low illumination and with previously undetectable objects, extensive experiments carried out in controlled laboratory conditions revealed notable increases in detection accuracy and robustness. Key results include a 98.6% Intersect over Union (IoU), an F1-score of 45.45% for object detection with DINO-DETR, and GPT-4.1 Vision precisely identifying 41 unique objects compared to 16 identified by MobileNetV4 and 6 by Detectron2. Our results highlight the possibilities of VLM-enhanced object detection in several robotic uses, including autonomous manipulation, disaster response, assistive technologies, inventory control, and surveillance.

1. Introduction

In recent years, autonomous robotic systems have greatly developed and shown growing capacity in environmental perception, navigation, and manipulation. One of the main difficulties still present in practical robotic applications, though, is the strong recognition and accurate identification of objects under different and erratic environments. Conventional object detection systems (e.g., Detectron2 and YOLO models) mostly depend on pre-defined, fixed training datasets (e.g., COCO), so restricting their capacity to identify and correctly classify new or out-of-distributed objects. Furthermore, these traditional methods usually fail under demanding visual conditions, especially in low-light situations and crowded surroundings. This work proposes an integrated robotic framework combining visual Simultaneous Localization and Mapping, autonomous path planning, soft robotic manipulation, and advanced object detection enhanced by Vision-Language Models, specifically GPT-4.1 Vision, so addressing these limitations. Our approach greatly improves object detection accuracy, contextual understanding, and robustness by using the zero-shot classification capabilities inherent in VLMs. Our robotic framework uses a modular quadruped fitted with an RGB-D camera using RTAB-Map for real-time visual SLAM without further odometry sensors. Using ROS Navigation Stack, the robot independently negotiates its surroundings and dynamically adjusts to changes in the surroundings and obstacles. Once at the target, the robot uses Detectron2 for initial object detection. GPT-4.1 Vision validates detections, corrects misclassifications, and uses semantic reasoning to identify new objects for frames with uncertain detection results. The effectiveness of our proposed system is validated through extensive experimentation in a controlled robotics laboratory environment, highlighting its potential in diverse practical applications including autonomous manipulation, disaster response, assistive robotics, inventory management, and surveillance.

2. Related Work

2.1. Simultaneous Localization and Mapping

Neural scene representations have lately been embraced in SLAM research for enhanced mapping and tracking. While ESLAM [21] combines NeRF-inspired architectures to mix accurate tracking with real-time dense mapping, NICE-SLAM and Co-SLAM [16] use implicit neural encoding to attain high-fidelity RGB-D reconstructions. Simultaneously, systems such as vMAP [8] add object-level neural representations inside SLAM maps, so enabling

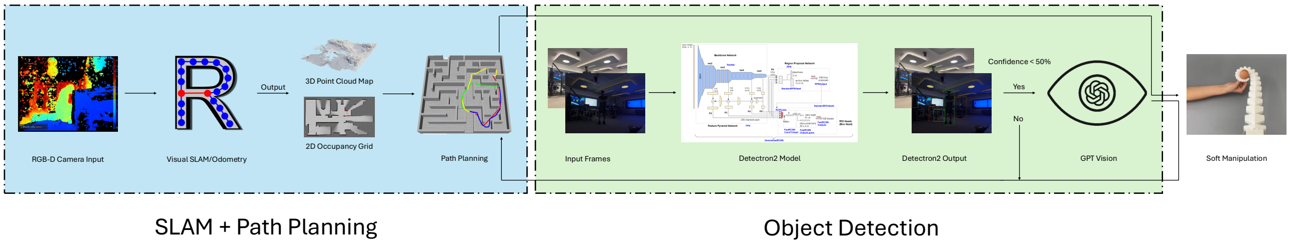

Figure 1. Overview of The Proposed Robotic Framework

scene understanding with instance awareness. By means of distributed back-ends and learnt loop closure mechanisms, collaborative SLAM systems such LAMP 2.0 [5] support multi-agent mapping. Driven by learnt scene representations and real-time dense mapping, these developments clearly show a turn towards semantic and multi-agent SLAM.

2.2. Path Planning for Mobile and Legged Robots

Modern path planning combines learning-based improvements [12] with classical algorithms. Especially for legged platforms, differentiable planners and transformer-based models have made end-to- end learning of feasible paths from visual inputs possible. Systems such as ArtPlanner [17] mix learnt heuristics with sampling-based motion generation for safe and effective navigation on uneven terrain for quadrupeds. Nowadays, hybrid solutions that combine learning with planning provide both strong performance and flexibility, which qualifies for dynamic and challenging surroundings.

2.3. Soft Manipulation

New gripper designs and sophisticated control techniques are driving ongoing evolution in soft robotic manipulation. To handle a range of object sizes and materials, hybrid grippers like RISOs [7] combine rigid and soft elements. Eliminating the need for internal sensors, methods such as Visual Force/Torque Sensing (VFTS) [1] estimate force feedback from gripper deformations using external cameras and deep learning. These advances greatly increase the robot's capacity for delicate and efficient interaction in unorganised surroundings.

2.4. Object Detection

Both CNN-based and Transformer-based models have advanced significantly in object detection. While DETR-based [20] models streamline detection pipelines through end-to-end training, one-stage CNN detectors such as YOLOv7 [15] deliver real-time performance with great accuracy. Modern results on COCO [11] benchmarks are obtained by combining deformable attention and query denoising in improvements like DINO and Mask DINO [10]. These contemporary detectors give the strong, fast object recognition required for real-time robotic uses.

3. Method

The proposed system integrates real-time mapping, autonomous navigation, and semantic object manipulation into a unified robotic framework. The robot uses an RGB-D camera and the RTAB-Map [9] library to do visual SLAM and visual odometry free from LiDAR or wheel encoders. This method produces a 2D occupancy grid, a constantly updated 3D point cloud, and preserves real-time robot location inside its surroundings. The user chooses a target destination on the occupancy grid once the environment is mapped. The system avoids obstacles by computing the best path from the robot's present position to the target using traditional path planning techniques including **Dijkstra** [2] or **A*** [3]. Once at the goal, the robot finds the target object using the Detectron2 [13] object detection system. The image is forwarded to a Vision-Language Model (GPT-4.1 Vision) [14] through API for validation or improvement if the detection confidence is lower than a specified threshold $< 50\%$, otherwise. A 3D-printed soft robotic manipulator mounted on the quadruped interacts with and grasp the object using the final object location as guide. Figure 1 shows the whole system parts together.

3.1. Simultaneous Localization and Mapping

This project applied RTAB-Map (Real-Time Appearance-Based Mapping) library inside Robot Operating System (ROS) architecture. Using a graph-based SLAM technique, RTAB-Map lets a robot create a map of its surroundings and concurrently localize itself within that map. Large-scale environments and appearance-based loop closure detection especially benefit from it. Our system uses an RGB-D camera as the only sensor providing depth information as well as visual data. While the depth data enables accurate three-dimensional mapping without additional sensors like LiDAR or wheel odometry, the RGB image offers rich visual characteristics for appearance-based localization. RTAB-Map handles the RGB-D data to estimate poses, match, and extract features. Keyframes are chosen to maximize computational efficiency depending on motion and appearance thresholds. Revisiting previously mapped areas reveals loop closures, which helps to correct accumulated drift in the trajectory via global graph optimization. Our system depends just on visual odometry derived from the RGB-D camera; no external odometry that from wheel encoders or IMUs is fused into the SLAM process in our configuration. This system shows the viability of low-cost, vision-based SLAM in indoor environments where conventional odometry sources could be absent or erratic.

3.2. Path Planning

The ROS Navigation Stack is used in this project to handle path planning both globally and locally, so enabling the robot to be safely moved from its present location to a chosen goal. The path planning process starts once the 2D occupancy grid and the location of the robot are acquired from the RTAB-Map SLAM module and a goal posture is chosen through RViz. Two kind of plans are produced by the navigation stack: a local and a global one. Combing a stationary, obstacle-free path from the robot's current position to the goal position falls to the global planner. Based on the known occupancy grid map, this route is computed just once upon goal setting. It avoids stationary objects and finds the shortest practical path. Operating in real-time, the local planner constantly changes the motion of the robot to consider dynamic obstacles and minor misallocation mistakes. Following the global road, the local planner creates velocity commands to securely negotiate unanticipated challenges and guarantee seamless navigation. Together, the local and global planners let the robot dynamically and firmly move towards the target in an always changing surroundings.

3.3. Object Detection

Detectron2 is an open-source object detection system created by Facebook AI Research (FAIR) used in the object detection module. Built on PyTorch, Detectron2 offers a modular, scalable, high-performance implementation of many state-of- the-art detection models. Based on RGB input frames obtained from the robot's on-board camera, Detectron2 in our system is in charge of spotting and localizing target objects inside the scene. Detectron2 fundamentally employs a two-stage detection pipeline based on Faster R-CNN architecture. The first stage creates candidate object bounding boxes from a Region Proposal Network (RPN) scanning the input image. The second stage polishes these proposals and assigns them particular object categories. Predicting objectness scores and bounding box coordinates across a sliding window on a convolutional feature map produces RPN proposals:

$$s = \sigma(f_{\text{cls}}(x)), \quad b = f_{\text{reg}}(x) \tag{1}$$

where f_{cls} and f_{reg} are the classification and regression heads, respectively, and σ is the sigmoid activation.

$$\hat{y} = \arg\max_{c \in \mathcal{C}} P(c \mid b, I) \tag{2}$$

where $\mathcal{C}$ is the set of object classes, b is the bounding box, and I is the input image. We configure Detectron2 to operate with a confidence threshold of 50%. If the detection confidence $< 50\%$ the frame is flagged as uncertain. We integrate GPT-4.1 Vision, a multimodal Vision-Language Model (VLM) created by OpenAI, to improve object detection robustness in uncertain or low-confidence settings. Key constraints discovered in conventional object detectors such as Detectron2 motivate the inclusion of GPT-4.1 Vision into our system. These models cannot identify unique or rare objects not included in their training data since they are naturally limited to a fixed set of class labels (e.g., COCO dataset). By using large pretraining on image-text pairs, GPT Vision can act as a zero-shot classifier recognizing and reasoning about a broad spectrum of objects without explicit retraining. Moreover, its semantic awareness helps it to remain useful in visually compromised situations, including low-light or occluded environments, where traditional detectors sometimes fail. In robotics, where systems must run autonomously in dynamic, unpredictable, or poorly lit real-world environments, these skills are especially important. GPT Vision thereby improves the robot's perception pipeline's adaptability, robustness, and contextual awareness. GPT-4.1 Vision leverages pre-trained knowledge across vision and language domains to reason about visual content, correct misclassifications, and identify previously unseen objects or categories.

4. Experimental Setup

4.1. SLAM

The robot was fitted with an RGB-D camera fixed firmly to provide the RTAB-Map SLAM system both visual and depth information for the SLAM experiments. We started by running RTAB-Map with just the RGB-D camera input, free of any additional odometry or inertial sensors, so enabling the library to manage both localization and mapping in whole 3D space. The accompanying Figure 3 clearly show that the first results revealed notable errors in both localization and mapping. Regarding localization, the projected robot trajectory displayed obviously erroneous behavior: the robot seemed to fly above the ground and move beneath the surface. The main cause of this mistake is the intrinsic noise in the depth measurements of the RGB-D camera, particularly under changing distances and lighting conditions. The produced 3D map also suffered with distortions. Parts of the map were wrongly projected below the ground level, and the rebuilt surroundings seemed slanted and uneven. These relics confirm even more the difficulties of depending just on noisy RGB-D data without the support of other sensing modalities like LiDAR or wheel odometry for pose stabilization.

SLAM Improvements

We changed the RTAB-Map configuration to limit the robot's motion to a planar case in order to solve the important problems of the slanted map and the inaccurate localization. We specifically limited the projected motion to allow just translations along the x and y axes and rotations around the z-axis (yaw). Given the robot moves in a flat, planar indoor environment, this assumption is reasonable. The mapping and localization outcomes much improved once planar-only motion was enforced. The updated Figure 5 show that the robot's projected path stayed on the ground plane and that the produced map no longer displayed strong tilting. A little vertical offset in the map persisted, but this is mostly ascribed to the robot's physical height above the ground about 0.5 meters.

4.2. Path Planning

We carried out path planning experiments using the 2D occupancy grid and the estimated position derived from the RTAB-Map SLAM system. RViz let one manually mark a goal location on the map. After choosing the target, the ROS Navigation Stack started the planning process to find a path from the robot's present position to the goal avoiding known obstacles. The blue line in Figure 6 shows the produced global path that which defines the stationary, obstacle-free path across the environment. Concurrent with this, the local planner constantly creates a dynamic path the yellow line to accommodate sensor-based trajectory corrections and real-time obstacle avoidance. Additionally seen on the map as a coordinate frame marker are the robot's current position and orientation. This configuration effectively showed how the combined SLAM and path planning system let the robot to independently negotiate the surroundings.

4.3. Object Detection

We explored several state-of- the-art object detection models in order to assess the efficiency of object detection in our robotic pipeline. Among the tested models are Detectron2, YOLOv8 [6], YOLOv11 [18], DINO-DETR [19] trained on COCO dataset. All models were set to a minimum confidence score of 25%. We also assessed a lightweight image classification model, MobileNetV4 [4], to offer a relative baseline for classification accuracy but it does not use localization like object detectors.

Dataset

Every model was tested on real-world RGB images taken straight from the RGB-D camera mounted on our quadruped robot. The dataset was gathered in a robotics laboratory with a range of object kinds, lighting settings, and scene complexity. This system was intended to replicate reasonable difficulties in autonomous perception including background clutter, varying brightness, and occlusion.

Evaluation Metrics

For evaluation, we adopted the following standard performance metrics:

- **Intersection over Union (IoU):** to assess the spatial accuracy of predicted bounding boxes
- **Precision:** the proportion of true positive detections among all positive predictions
- **Recall:** the proportion of true positives among all actual instances of an object
- **F1 Score:** The harmonic mean of precision and recall, providing a balanced metric especially useful under class imbalance.

Baseline

The YOLOv8x model served as the baseline for all comparisons, Because of its good performance in real-time detection tasks and its efficiency across many hardware platforms

4.4. Computational Resources

All experiments were run on a workstation with an NVIDIA RTX 3080 GPU, Intel Core i5-14600KF processor, and 32 GB of DDR5 RAM, ensuring fast and consistent training and inference across all tested models.

5. Results & Discussion

5.1. SLAM Results

The effects of restricting the motion of the robot to a planar model are shown by means of the SLAM experiment outputs. Noise in the RGB-D camera measurements caused great inaccuracy in the localization and mapping outputs first, when allowing complete 3D motion estimation. Figures 2 and 3 showed that the generated map displayed severe tilting and distortions while the estimated trajectory of the robot seemed to move unrealistically above and below the ground.

The results much improved once the RTAB-Map arrangement was changed to limit motion to the planar case (x, y translation and rotation around z-axis). Figures 4 and 5 show that the estimated path of the robot stayed on the ground plane and that the map became notably more aligned and stable. The robot's physical height left a small vertical offset in the map, but generally mapping accuracy was much improved.

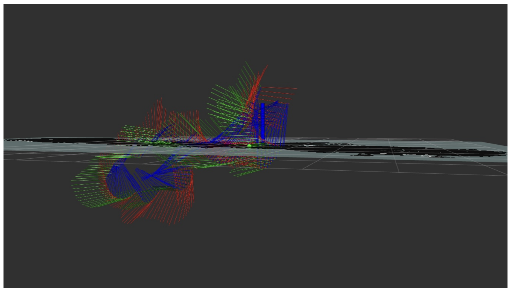

Figure 2. Localization result before enforcing planar motion, showing unrealistic vertical motion.

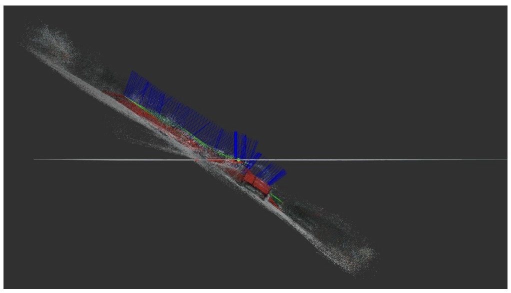

Figure 3. Mapping Result Before Enforcing Planar Motion, Showing a Tilted and Distorted Map.

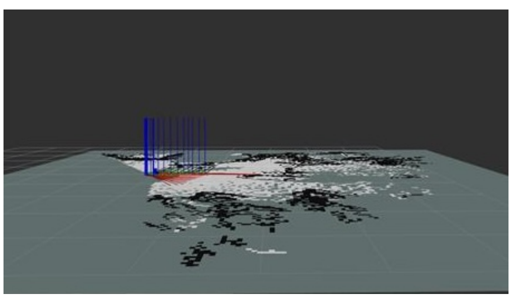

Figure 4. Localization Result After Enforcing Planar Motion, Showing Consistent Movement on The Ground Plane.

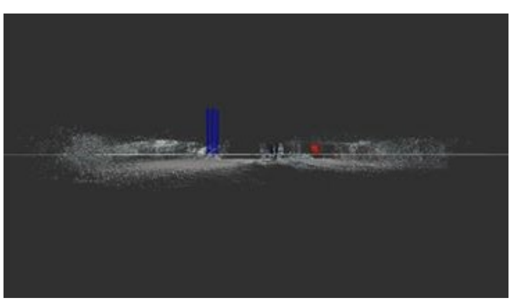

Figure 5. Mapping Result After Enforcing Planar Motion, Showing an Aligned and Stable Map.

5.2. Path Planning Results

From the robot's present position to several goal positions indicated on the map, the path planning experiments showed effective navigation. While the local planner dynamically changed the robot's trajectory in real-time to avoid any obstacles and guarantee smooth motion, the global planner produced a stationary, obstacle-free path (visualized as the blue line) upon setting a goal through RViz. The robot could precisely follow the created paths and reach the targets free from collision. Figure 7 shows the local path, global path, and robot position under navigation. While the local path constantly changes as the robot traverses the surroundings, the figure emphasizes how the global path directs the overall course.

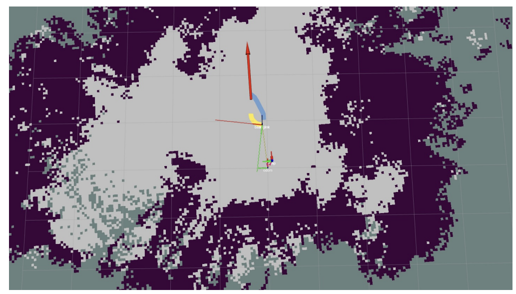

Figure 6. Global Path (Blue Line), Local Path (Yellow Line), and The Robot's Current Location (Coordinate Frame) Visualized During Navigation in RViz.

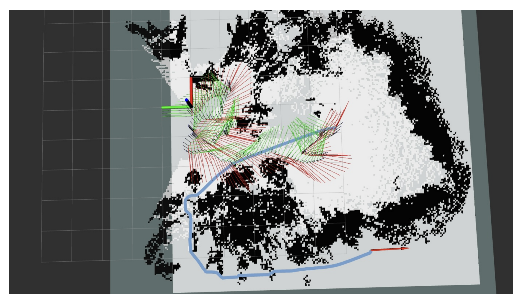

Figure 7. Global Path (Blue Line) Shown Avoiding Obstacles.

5.3. Object Detection

Model	IoU (%)	Inference Time (s)	Avg Confidence (%)	Precision	Recall	F1 Score
Yolov8x	91.6	1.27	53	57.14	**32.0**	41.03
Yolov11x	97.3	1.32	50	75.0	18.18	29.27
DINO-DETR	**98.6**	**1.94**	74	90.91	30.3	**45.45**
Detectron2	91.5	1.88	**83**	**92.7**	24.24	39.02

Table 1. Comparison of Object Detection Models on Various Metrics in a Robotics Lab Environment.

Despite its longer inference time, DINO-DETR emerged as the top-performing model, achieving the highest IoU

(98.6%) and F1 score (45.45). This superior performance can be attributed to its transformer-based architecture and end-to-end optimization, which enable it to more effectively model global context and complex object relationships. Detectron2 showed the lowest recall (24.24%), but the highest average precision (92.7%) and confidence (83%). This indicates it was quite accurate when it did detect objects, but often failed to detect many instances likely due of its reliance on region proposals that can miss smaller or occluded objects. While obtaining high IoU and precision, YOLOv11x had the lowest recall (18.18%), reflecting a similar restriction in coverage, maybe caused by aggressive confidence thresholds or insufficient feature representation for various object scales. More suited for real-time applications, YOLOv8x selected as the baseline showcased the best trade-off between speed (1.27s inference time) and consistent detection performance. Ultimately, while DINO-DETR provides the best detection accuracy, its latency makes it less practical for real-time use, justifying our selection of Detectron2, which offers a balanced compromise across inference time, spatial accuracy, and reliability for our proposed framework 3.

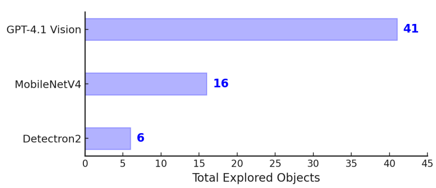

Figure 8. Number of Objects Identified by the Object Detector, Classifier and VLM.

With 16 objects successfully recognized, MobileNetV4 outperformed Detectron2 as shown in Figure 8, which could only identify 6. This demonstrates the drawback of Detectron2's dependence on a fixed training dataset (COCO), which only has 80 pre-defined object categories. GPT-4.1 Vision Compared to MobileNetV4 (16 objects) and Detectron2 (6 objects) was able to identify 41 different objects in total. The zero-shot capability of vision-language models, which allows them to recognize a large variety of objects even if they were not specifically included in any training dataset, is strongly supported by this result.

In low light conditions, Detectron2 performs worse as shown in Figure 9. Detectron2 demonstrated satisfactory accuracy and confidence in detecting and labelling objects in well-lit scenes. But when the illumination decreased, the model either misclassified known categories, failed to identify important objects, or generated inaccurate bounding boxes. These findings point to a significant drawback of conventional object detection models: their reliance on steady illumination and sharp vision is crucial. This sensitivity to lighting becomes a significant concern for real-

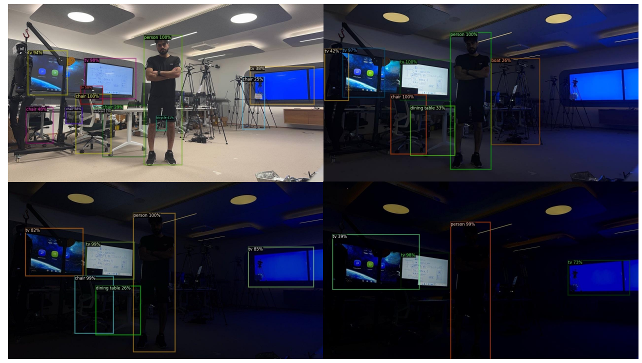

Figure 9. Different Light Conditions Tested Using Detectron2

world robotic applications, particularly those that operate in dim environments like warehouses, nighttime settings, or disaster zones. These results strengthen the case for adding vision-language models to the pipeline of perception. The darkest fourth image shown in Fiure 9 was passed to GPT-4.1, below shows the user prompt and the GPT Vision response.

Prompt: *This is an image used for Object Detection Model. List all the objects and the things seen in the picture.*

Result: *The image is quite dark, but the following objects are visible: Monitors, Chairs, Desk/Table, Ceiling Lights, Person, Tripod/Camera, Laptop/Computer, Window, Plant.* As shown in the previous Prompt/Response, GPT Vision was able to identify a wide range of objects, including monitors, chairs, a desk or table, ceiling lights, a person, a tripod or camera, a laptop, a window, and even a plant, even in the presence of low illumination, an area where traditional vision models frequently fall short. The zero-shot classification capability of GPT Vision is demonstrated by the fact that a number of these objects, including the tripod and plant, are not included in the standard COCO dataset that is used to train detectors like Detectron2. Even in visually impaired environments, GPT Vision uses semantic reasoning and natural language understanding to infer object identities, in contrast to traditional models that are limited by fixed label sets and necessitate strong visual cues. In robotic vision applications where environments are unstructured, variable, and unpredictable, this highlights two key benefits: (1) robustness to poor lighting, and (2) the ability to recognize novel objects without retraining. The integration of Vision-Language Models into robotics pipelines to facilitate more intelligent, context-aware perception is strongly supported by these findings.

Prompt: *This is a result picture of an Object Detection Model. How many bounding boxes are there? Are these objects inside the bounding boxes classified correctly? What about other objects? What are they?*

Result: *The image contains 12 bounding boxes with de-*

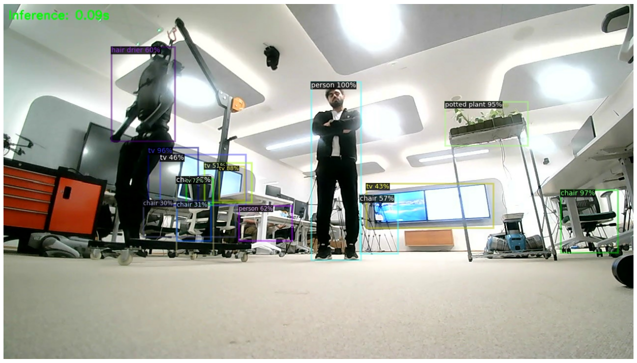

Figure 10. Detectron2 Output Frame From Quadruped Robot Camera

tected objects like TVs, chairs, a person, and a potted plant. Most are correctly labeled, but the "hair drier" is likely a misclassification (possibly a mannequin or equipment). Some objects like desks, cabinets, and floor equipment are visible but not detected.

We provided GPT-4.1 Vision with a result image from Detectron2 including several object detections as shown in Figure 10 in order to evaluate its semantic reasoning capacity. The prompt asked GPT-4.1 Vision to find any extra, unnoticed objects as well as to evaluate the accuracy of classifications and examine the bounding boxes. Accurately identifying 12 bounding boxes and correctly labelled objects including TVs, chairs, a person, and a potted plant, the model responded. Especially, GPT Vision pointed out a misclassification whereby a "hair drier" label was probably false and recommended using another kind of equipment, such a mannequin, which is correct. By means of zero-shot classification and contextual reasoning, the model also effectively detected undetectable objects including desks, cabinets, and floor equipment, so augmenting conventional object detectors. Not only for object recognition but also for verifying the accuracy of current detections and offering richer scene understanding in challenging environments, this emphasizes the need of including Vision-Language Models into perception pipelines.

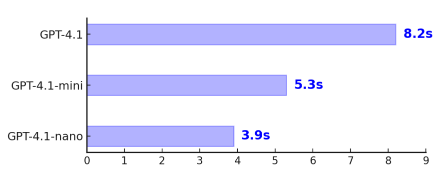

Figure 11. Processing Time Comparison of a Different GPT-4.1 Vision Variants.

Three variants of the GPT-4.1 Vision model GPT-4.1, GPT-4.1-mini, and GPT-4.1-nano have their processing speed compared in Figure 11. The longest processing time (8.2 seconds), followed by GPT-4.1-mini (5.3 seconds), and GPT-4.1-nano, which offered the fastest inference time (3.9 seconds), was expected of the standard GPT-4.1 Vision model. These findings clearly emphasize the natural trade-off among model complexity, accuracy, and processing speed. While larger variants like GPT-4.1 typically offer higher zero-shot accuracy and deeper contextual understanding, they come at the cost of longer inference times, which could limit their use in latency-sensitive robotic applications. On the other hand, GPT-4.1-nano may compromise some recognition robustness and semantic detail while yet providing much faster inference fit for real-time situations.

6. Limitations

- **Depth Noise Sensitivity** RGB-D camera introduces noise at varying distances and lighting, causing mapping distortions and localization drift.
- **No Multi-Sensor Fusion** System relies solely on visual odometry - no IMU or encoder data for pose correction under poor visual conditions.
- **Planar Environment Constraint** Assumption of a flat surface (x, y, yaw) improves map quality but limits use in multi-floor or sloped environments.
- **Loop Closure Failures** In flat or repetitive environments, or when the robot moves too fast, loop closures may fail to trigger, reducing SLAM accuracy and causing trajectory drift.
- **Inference Latency in Larger VLMs** With a fairly high processing time (8.2 seconds per frame), the standard GPT-4.1 Vision model might not be appropriate for real-time robotic applications where fast reaction times are crucial. Smaller variants (mini and nano) may trade off semantic richness and zero-shot recognition accuracy even though their latency was much lowered.
- **Hallucinations and Overgeneralization** GPT-4.1 Vision performed well in low-light environments and zero-shot classification, but in rare cases or poorly written prompts it could hallucinate false object labels or inferred nonexistent objects. These false positives in absence of strong bounding box priors or confidence filtering could result in erroneous downstream robotic manipulation decisions.
- **Limited Evaluation in Outdoor and Dynamic Environments** Every experiment was carried out indoors under carefully regulated lighting and layout. The generalisability of present results is limited by the lack of evaluation of the system's performance in outdoor, fast changing, or highly dynamic environments (e.g., rain, sunlight, crowds).
- **Computational Cost and Energy Consumption** Running high-capacity VLMs on embedded or mobile robotic platforms could be impractical in terms of GPU resource demands and energy consumption, so posing problems

for field robotics with limited power.

7. Conclusion

We presented in this work a modular robotic framework including visual SLAM, autonomous path planning, soft robotic manipulation, and object detection improved by GPT-4.1 Vision. Particularly under low-light and with new objects, our method greatly enhanced object detection robustness, accuracy, and semantic understanding. GPT Vision's zero-shot features especially extended detection beyond conventional dataset limits. High latency, occasional hallucinations, limited outdoor evaluation, and computational requirements are among the system's challenges. Future work will Incorporate multi-sensor for SLAM. For Path Planning, we will try Learning-based planners. In Object Detection, we will concentrate on optimizing model efficiency for real-time performance, improving dependability by means of advanced filtering and prompt engineering, assessing performance in dynamic outdoor environments, and integrating totally autonomous soft manipulation to maximize real-world applicability.

References

[1] Jeremy A. Collins, Patrick Grady, and Charles C. Kemp. Force/torque sensing for soft grippers using an external camera. *arXiv preprint arXiv:2210.00051*, 2022. 2

[2] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959. 2

[3] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. In *IEEE Transactions on Systems Science and Cybernetics*, pages 100–107. IEEE, 1968. 2

[4] Andrew Howard, Mark Sandler, et al. Mobilenetv4 (unofficial). https://github.com/google-research/google-research/tree/master/mobilenetv4, 2024. Accessed: 2025-04-18. 4

[5] Alexander Hughes and Andrew Calway. LAMP 2.0: Multi-Session SLAM with Object-Level Mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. Available at: https://www.slamcore.com/news/visual-slam-research-papers-iros-2022. 2

[6] Glenn Jocher, Ayush Chaurasia, Tao Qiu, Laughing Stoken, et al. Yolov8: Cutting-edge object detection models. https://github.com/ultralytics/ultralytics, 2023. Accessed: 2025-04-18. 4

[7] Maya Keely, Yeunhee Kim, Shaunak A. Mehta, Joshua Hoegerman, Robert Ramirez Sanchez, Emily Paul, Camryn Mills, Dylan P. Losey, and Michael D. Bartlett. Combining and decoupling rigid and soft grippers to enhance robotic manipulation. *arXiv preprint arXiv:2404.13755*, 2024. 2

[8] Lingzhi Kong, Yisheng He, Lingjie Liu, and Christian Theobalt. vMAP: Vectorised Object Mapping for Neural Field SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11273–11285, 2023. 1

[9] Mathieu Labbé and François Michaud. Rtab-map: Real-time appearance-based mapping. https://introlab.github.io/rtabmap/, 2019. Accessed: 2025-04-18. 2

[10] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M. Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20367–20377, 2023. 2

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *arXiv preprint arXiv:1405.0312*, 2014. 2

[12] Jianwei Liu, Shirui Lyu, Denis Hadjivelichkov, Valerio Modugno, and Dimitrios Kanoulas. Vit-a*: Legged robot path planning using vision transformer a*. *arXiv preprint arXiv:2310.07525*, 2023. 2

[13] G. M. Merz, Y. Liu, C. J. Burke, P. D. Aleo, X. Liu, M. C. Kind, V. Kindratenko, and Y. Liu. Detection, instance segmentation, and classification for astronomical surveys with deep learning (deepdisc): Detectron2 implementation and demonstration with hyper suprime-cam data. *arXiv preprint arXiv:2307.05826*, 2023. 2

[14] OpenAI. Gpt-4 technical report. https://openai.com/research/gpt-4, 2024. Accessed: 2025-04-18. 2

[15] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022. 2

[16] Wenxuan Wang, Yisheng He, Lingjie Liu, and Christian Theobalt. Co-SLAM: Joint Coordinate and Sparse Parametric Encodings for Neural Real-Time SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18984–18994, 2023. 1

[17] Lorenz Wellhausen and Marco Hutter. Artplanner: Robust legged robot navigation in the field. *arXiv preprint arXiv:2303.01420*, 2023. 2

[18] Wong Kin Wong. Yolov11: Real-time object detection with next-generation improvements. https://github.com/WongKinYiu/yolov11, 2024. Accessed: 2025-04-18. 4

[19] Feng Zhang, Xizhou Wang, Jifeng Lv, Kai Huang, Wengang Li, Ping Luo, Yichen Yao, and Jian Sun. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 4

[20] Yingying Zhang, Chuangji Shi, Xin Guo, Jiangwei Lao, Jian Wang, Jiaotuan Wang, and Jingdong Chen. Enhancing detr's variants through improved content query and similar query aggregation. *arXiv preprint arXiv:2405.03318*, 2024. 2

[21] Zihan Zhu, Songyou Peng, Martin Larsson, Michael Bloesch, and Angela Dai. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12786–12796, 2022. 1