



Q-Learning with VLA Perception for Real-World Navigation Using Low-Cost Sensors

Madhuri Manohar Barhate¹ , Aditya K. Vasipalli¹ ,
Anushri Arora¹ , Kaivalya Beohar¹ , Param Bande¹ 
, Aaysha Khan¹ 

¹CSE AIML, Vishwakarma Institute of Technology, Pune, India
adityavasipalli@gmail.com

Abstract. Vision-Language-Action (VLA) models allow robots to understand scenes alongside linguistic instructions. These models have computational requirements that are too demanding to implement on edge hardware like a Raspberry Pi and result in an unacceptable latency gap between the interpretation and decision-making stages. Despite GPU acceleration or remote inference, the latency problem still persists, creating delays between perception and action that make closed-loop control systems unstable and unsafe. This paper provides an implementation of a Slow-Fast Hybrid Control Architecture in order to be able to deploy foundation models on low-cost edge devices with remote inference. The slow system running on the cloud uses a highly quantized VLA model that detects objects and gives navigation instructions. The fast system running on the Raspberry Pi uses a Reinforcement Learning policy to provide motor commands in real time, and the RL agent is rewarded using a multi-objective reward function guided by the VLA instructions.

Keywords: Vision-Language-Action Models, Reinforcement Learning, Edge Robotics, Low-Cost Navigation, Latency-Aware Control, Hybrid Architecture

1 INTRODUCTION

The integration of Large Language Models (LLMs) with visual encoders resulting in Vision-Language Models (VLMs) and subsequently Vision-Language-Action (VLA) models has brought forth a new era in robotics. Unlike the conventional robotics approaches where the pipeline requires pre-defined categories of objects and rigid state machines, the Vision-Language-Action (VLA) approaches can perceive natural language commands like "go to the red cup on the left of the laptop" and perform actions based on these commands in an unstructured environment. Some recent VLA models like RT-2, OpenVLA, and $\pi 0$ have been successful at demonstrating remarkable zero-shot generalization abilities in many different tasks. However, there is still a significant gap between these abilities of the models and what robots in the real world actually need. Robots need high-frequency control loop mechanisms to avoid odometry errors and respond to dynamic obstacles.

In contrast, the computational cost of utilizing VLA models is extremely high. The open-source models, like LLaVA-7B, necessitate large matrix multiplication operations that are challenging to perform with commercial GPUs. This task becomes more complicated if such models are executed on embedded systems

without server inference because of latency and bandwidth problems. In many cases, especially with low-cost robotics, platforms such as Raspberry Pi 4 or other single-board computers with ARM Cortex-A72 processors are often used. Unfortunately, such devices do not have hardware accelerators like CUDA cores or Tensor Processing Units (TPUs). Thus, the execution of VLA models on such devices takes much more time than it could be done on GPUs or TPUs. The execution of a 7-billion-parameter model can take several seconds per inference. It is especially important in cases like precision agriculture when robots should work autonomously in a vast unstructured environment with low connectivity and limited computational capabilities.

1.1 Frequency Mismatch Problem

The main technical problem which this paper solves is the problem of frequency mismatch. When a robot depends only on the VLA model for its control, the limitation of the speed of the model inference imposes constraints on the speed of the control loop. The robot will therefore be executing a certain action for several seconds before it gets the next control command. In practice, this would cause collisions, accumulation of odometry errors, and decreased success rates of the tasks performed, especially in dynamic environments. Current solutions to this problem include offloading computation and using small models. Offloading computation introduces dependence on network connectivity and variable latency, while small models do not provide semantic reasoning and generalization capabilities that are the main advantages of VLA models.

1.2 Proposed Solution: Latency-Tolerant Hybrid Control

We design an architecture called Hybrid VLA+RL Architecture that features two distinct components, namely semantic reasoning and hardware control. Building on Kahneman's dual-process theory (System 1 vs. System 2) and recent implementations of dual-system foundation models [7], our proposed architecture is:

- **System 2 (Slow):** A quantized LLaVA-7B model running on a server with a GPU. It processes visual scenes to generate high-level semantic waypoints.
- **System 1 (Fast):** A localized Reinforcement Learning (RL) policy running at 50Hz. This policy takes the waypoint from System 2, combines it with real-time Ultrasonic sensor data, and generates motor command.

1.3 Contributions

- Hybrid slow–fast architecture for control under delayed VLA perception
- Latency-aware Q-learning formulation incorporating τ in state and reward
- Mode-based adaptive control (explore, search, navigate) under stale perception
- Real-world validation on low-cost robotic hardware

2 Literature Review

Three main lines of research have shaped the development of embodied intelligence: scaling multimodal foundation models [1], optimizing these models for edge devices with limited resources [2], and creating hierarchical architectures that solve the problem of reasoning and reactive control happening at the same time [7].

2.1 Generalist Vision-Language-Action (VLA) Models

The move towards generalist robots started with the introduction of models like Google's RT-2, which demonstrated that large-scale pretraining on internet-scale data greatly increases the robot's ability to understand novel language commands and generalize to unseen objects [1]. Based on this concept, OpenVLA was introduced as a leading open-source model [3]. The model consists of the Llama 2 backbone and fused visual encoder based on DINOv2 and SigLIP features, making the fine-tuning of generalist policies possible using the Open X-Embodiment dataset consisting of almost one million robot trajectories [3]. More recent benchmarking has compared OpenVLA with other state-of-the-art models, such as RDT-1B and ACT [5]. In this context, it can be said that although generalist models perform well semantically, they struggle with stability in high-frequency tasks, such as bimanual manipulation.

2.2 Edge Optimization and On-Device Navigation

One of the major difficulties of using VLA models is the frequency mismatch problem, which means that the computational cost of model inference goes against the need of 50 to 100 Hz control frequency for stable robotics work [11]. Several techniques have been suggested to solve this problem in low-cost hardware like the Raspberry Pi 4. Relevant research works have also considered the application of foundation models in limited resource scientific and robotic automation tasks [6]. Some of these techniques include quantization, which includes GGUF 4-bit precision, and allows 7-billion parameter models to fit in the 8 GB RAM of single-board computers, but at the expense of increased inference latency [8]. EfficientNav solves the problem of planning latency through the elimination of redundant data in the navigation map through semantic memory retrieval and KV-cache clustering based on attention [9]. Furthermore, action chunking and parallel decoding have also been used to allow prediction of several future actions in one inference step [3].

2.3 Hierarchical and Dual-System Control Architectures

Inspired by Kahneman's dual-process theory, recent studies have started to focus more on System 1/System 2 architectures [7]. In such systems, the slower System 2 process consists of a VLA model that is responsible for reasoning and semantic planning, whereas the faster System 1 process deals with real-time reactive control [7]. The HiRT

system adopts such architecture by employing a vision-language model like InstructBLIP to provide latent features to the smaller and higher frequency execution policy [4]. Likewise, the FiS model incorporates the execution module with the final transformer layers of the foundation model, thus allowing parameter sharing with the asynchronous control frequency up to 117.7 Hz [7].

2.4 Applications in Precision Agriculture

Navigation and path tracking remain a challenge for robots working in an unstructured environment [13]. For the real-time detection of weeds and plant diseases, solutions like Agrova and the Leo rover make use of multimodal sensors combined with edge-based AI [14]. Agricultural foundation models (AFMs), which include Agri-LLaVA and AgroGPT, offer interactive interfaces for pest and disease diagnoses [15]. The Farm-LightSeek, a 1 billion parameter model based on LLaVA, showed that models at such scale can achieve inference times of about 680 ms on Raspberry Pi-class hardware, allowing autonomous treatment recommendations even in intermittent connectivity conditions [16]. Moreover, hierarchical reinforcement learning (RL) models such as HAM-PPO optimize the process through trade-offs between the infestation coverage and battery/pesticide consumption [17].

3 Theoretical Framework

3.1 State Representation

The control policy uses a highly discretized state space to keep the Q-table computationally tractable on the device itself. The state space is formally defined by the following 6-tuple:

$$s = (s_{sensor}, \theta_{vla}, d_{vla}, c_{vla}, m, \tau) \quad (1)$$

where:

- $s_{sensor} \in S_{sensor}^4$ represents discretized obstacle distances from ultrasonic sensors across four directional axes.
- θ_{vla} denotes the relative bearing to the semantic target, quantized into directional angular bins.
- d_{vla} represents the VLA-estimated physical distance, discretized into proximity zones.
- $c_{vla} \in [0,1]$ is the semantic confidence score generated by the VLA.
- $m \in \{1, 2, 3\}$ denotes the current discrete operational mode (1: Exploration, 2: Navigation, 3: Search).
- τ represents the latency feature, defined as the time elapsed since the last VLA update, discretized into temporal bins.

3.2 Reinforcement Learning Formulation

The fast control system is implemented using a tabular Q-learning approach, where the agent learns a mapping from states to actions through interaction with the environment.

The action space consists of discrete motion primitives

- forward movement
- left turn
- right turn
- stop

The policy is updated using the standard Bellman equation:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)] \quad (2)$$

where r represents the reward obtained after taking action a in state s , α is the learning rate ($\alpha = 0.1$) and γ is the discount factor ($\gamma = 0.95$).

Multi-Objective Reward Function

The reward function balances four competing objectives- progress toward the target, semantic alignment, obstacle avoidance, and movement efficiency, through the weighted sum:

$$r(s, a, s') = r_{step} + r_{obs}(s, a) + r_{eff}(s') + \varphi(\tau) \cdot r_{sem}(s, s') + r_{nav}(s') \quad (3)$$

where:

- $r_{step} = -0.1$ — small penalty per action.
- $r_{obs}(s, a) =$
 - -50 if front ultrasonic < 0.15 m(collisions),
 - -10 if front < 0.25 m(near obstacle),
 - 0 otherwise.
- $r_{eff}(s') = -0.05 \cdot (\text{movement_duration})$ — penalizes slow execution.
- $r_{sem}(s, s')$ semantic progress reward:
 - $+5$ if v transitions $0 \rightarrow 1$ (target acquired),
 - $+2$ if β becomes CENTER when previously not,
 - $+1$ if δ decreases (target closer),
 - -2 if v transitions $1 \rightarrow 0$ (target lost),
 - 0 otherwise.
- $r_{nav}(s') = +100$ when $(\beta = \text{CENTER} \wedge \delta = \text{NEAR} \wedge u = \text{CLEAR})$, otherwise 0 (goal reached). It signifies the goal reaching reward.
- $\varphi(\tau)$ is the VLA freshness gate, defined as:

$$\varphi(\tau) = \tanh\left(\frac{5}{\tau + 0.1}\right) \quad (4)$$

This smoothly reduces semantic reward influence as τ increases: $\varphi(\tau) \approx 1$ when $\tau < 3$ s, and $\varphi(\tau) \lesssim 0.3$ for $\tau > 15$ s.

3.3 Latency-Aware Decision Making

A crucial design aspect is the explicit use of τ in the state and reward, allowing the agent to modify its reliance on semantic information dynamically. The mode (m) evolves as follows:

Mode determination (m):

- NAV $\rightarrow$ if $v = 1$
The target is recently seen; Q-learning drives the robot toward the target using both semantic and local rewards.
- SEARCH $\rightarrow$ if $\tau_{stale} < \tau < \tau_{explore}$
No target in frame, but the *latency* is moderate. The agent reduces its trust in the last VLA bearing and begins a memory-guided rotation recalling ℓ (last seen direction). The $\varphi(\tau)$ gate halves the semantic reward weight.
- EXPLORE $\rightarrow$ if $\tau \geq \tau_{explore} = 15 \text{ steps } (\approx 19.5 \text{ s})$
The VLA data is expired. The agent falls back to an exploration pattern until the target is re-acquired. The semantic reward term is effectively zero.

A lightweight threshold sets mode m before action selection (and is stored in the state key), so the tabular Q-agent learns distinct policies per regime; this dual-system, fast local Q-controller with ultrasonic reflexes and slow VLA semantic updates, ensures responsive, robust behavior under delayed or stale perception.

3.4 High-Frequency Control Under Intermittent Perception

The architecture described in this work separates control execution from semantic reasoning and supports a signal flow that maintains the decision-making loop (or perception and action looping).

By leveraging continuous local sensor feedback, the RL controller maintains a (local) real-time decision-making ability, while at the same time the VLA module provides sparse, higher-level updates (non-real-time) to the RL controller. The combination of these two components allows for continuous interaction with the world even in the case of large perception delays (i.e., VLA-perception delays). In summary, the VLA functions to provide semantic guidance while the RL controller functions to provide real-time control of the system.

4 System Architecture

4.1 Hardware Configuration

Raspberry Pi 4 (4GB RAM), remote GPU (Nvidia Geforce RTX 4060), Robot hardware includes 2x TB6612FNG motor driver, 4x HC-SR04 ultrasonic sensors (front/left/right/back), and a USB camera.

4.2 Slow System: Remote Vision–Language Perception

- Model: LLaVA-v1.5-7B quantized to Q4 K M (4.08 GB).
- Engine: llama.cpp with OpenBLAS and ARM NEON.
- Prompting: Constrained JSON output reduces generation tokens and latency.

4.3 Fast System: Latency-Aware RL Agent

- Algorithm: Tabular Q-learning (Q-table).
- Input: Discretized ultrasonic obstacle state + semantic target cues (visibility, bearing, distance) with latency-aware mode.
- Output: Discrete motion primitives (forward, left, right, stop), with safety override.
- Execution: On-device hierarchical control (fast ultrasonic safety, mid exploration, slow goal-seeking Q-learning).

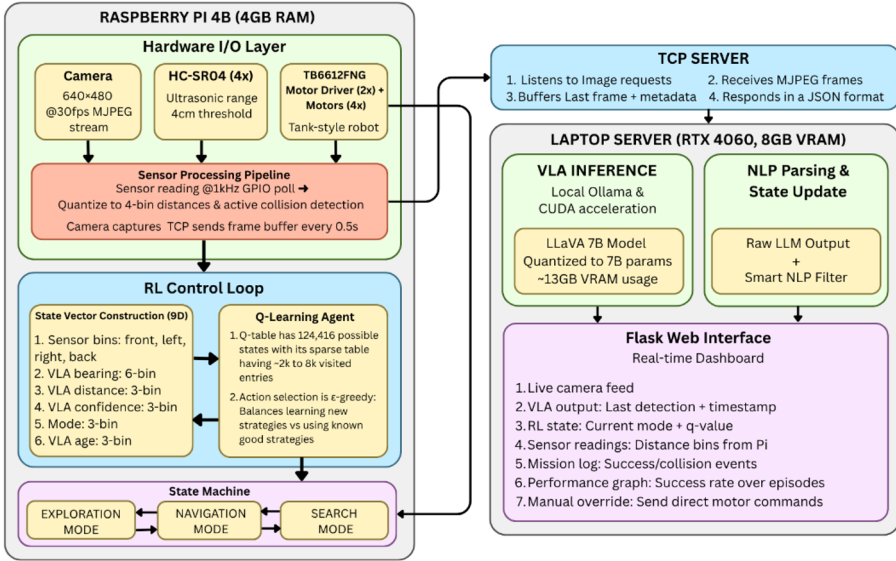


Fig. 1. Proposed hybrid VLA+RL system architecture.

5 Evaluation

Baselines:

- B0 Monolithic VLA: open-loop between inferences, contains frequency mismatch.
- B1 Classical: YOLOv5-Nano + A* + DWA.
- B2 Dual-System (No- τ): slow semantic planner + fast local controller, no latency feature, freshness gating is disabled.
- B3 Hybrid FiS (Ours): dual system + latency-aware state/reward using τ , adaptive mode-switching and multi-objective reward gating.

Results:

Table 1 shows the real world performance.

Table 1. Performance across methods

Metric	Monolithic VLA	Classical	Dual-System (No- τ)	Proposed
Success Rate	12.0%	60.0%	56.0%	69.2%
Collision Rate	66.0%	10.0%	18.0%	4.0%
Avg. Time to Goal	N/A	25 s	31 s	38 s
Success weighted by Path Length (SPL)	0.08	0.45	0.40	0.58

All tests were performed on a Raspberry Pi 4 using real-world navigation. Task: “Find and approach a green plant object.” In total, 65 randomized trials were performed for each approach. A trial was considered successful if the robot reached within 0.5 m of the target plant. When an ultrasonic sensor showed a distance less than 0.15 meters, collision was recorded.

Ablation Study:

Baseline validity and Effect of latency awareness:

- B0 (Monolithic VLA): Captures frequency mismatch failure, low reliability (SR =12.0%), high-risk setting (collision rate =66.0%).
- B1 (Classical baseline) : Increases stability (SR =60.0%), reduces collision (=10.0%) but is a task-specific engineered stack.
- The key ablation is between B2 and B3: Both are dual-system approaches, however, B2 does not employ latency awareness (τ), while B3 uses a freshness-aware controller. In this comparison, B3 shows improved SR performance (from 56.0% to 69.2%, +13.2 pp) and reduced collision (18.0% to 4.0%, -14.0 pp). It suggests that latency awareness is one of the significant components of the system, not a minor addition.
- The results also show a deliberate safety vs efficiency trade-off. Compared to B2, B3 increases average time-to-goal from (31) s to (38) s, but achieves safer behavior and better navigation quality.
- Relative to the classical baseline, B3 further improves SR (+9.2 pp), lowers collision (-6.0 pp), and yields higher path-efficiency (SPL: (0.58) vs. (0.45)).
- Taken together, these values indicate that incorporating τ -aware decision-making improves robustness under stale semantic updates, with a moderate cost in completion time that is acceptable for safety-critical operation on resource-constrained platforms.

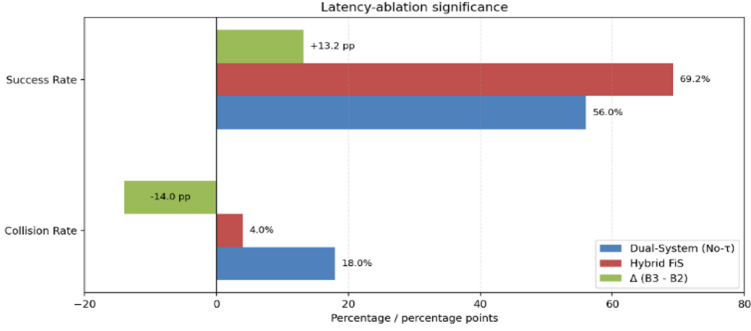


Fig. 2. Latency-ablation significance (Dual-System without τ vs Hybrid FiS with τ)

6 Conclusion

This work implemented a latency aware Vision-Language-Action + Reinforcement Learning robotic system using low cost hardware. The inference latency (~ 4 s) is addressed as a design constraint instead of using real time perception. Our approach involved integrating intermittent VLA semantic feedback alongside continuous Reinforcement Learning Control using Q-Learning, thus bridging the temporal gaps inherent in large scale multimodal models. Adaptive behavior is enabled by our three mode state machine i.e, exploration, navigation and search, while the real-time responsiveness during perception gaps is ensured by the RL agent. Experimental results demonstrate an improved success rate, highlighting the fact that real-time responsiveness is achievable as well as reliable in VLA driven robotic systems without continuous multimodal inference.

7 Declaration

Funding: The authors did not receive support from any organization for the submitted work. **Competing Interests:** The authors have no relevant financial or non-financial interests to disclose. **Ethics Approval:** Not applicable. This study does not involve human participants or animals. **Data Availability:** The datasets analyzed during the current study are available from the corresponding author on reasonable request. **Author Contributions:** All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by Madhuri Manohar Barhate, Aditya K. Vasipalli, Anushri Arora, Kaivalya Beohar, Param Bande, and Aaysha Khan. All authors read and approved the final manuscript.

References

1. Sapkota, R., Cao, Y., Roumeliotis, K.I., Karkee, M.: Vision-language-action (VLA) models: Concepts, progress, applications and challenges. arXiv:2505.04769 (2025). <https://arxiv.org/html/2505.04769v2>

2. Zhang, D., et al.: Pure vision language action (VLA) models: A comprehensive survey. arXiv:2509.19012 (2025). <https://arxiv.org/abs/2509.19012>
3. Kim, M.J., et al.: OpenVLA: An open-source vision-language-action model. In: Proc. Mach. Learn. Res., vol. 270, pp. 2679–2713 (2025). <https://doi.org/10.48550/arXiv.2406.09246>
4. Zhang, J., et al.: HiRT: Enhancing robotic control with hierarchical robot transformers. In: Proc. Mach. Learn. Res., vol. 270, pp. 933–946 (2025). <https://proceedings.mlr.press/v270/zhang25b.html>
5. Zhang, Y., Qi, Y., Zheng, X.: Experiences from benchmarking vision-language-action models for robotic manipulation. arXiv:2511.11298 (2025). <https://arxiv.org/html/2511.11298v1>
6. Hatakeyama-Sato, K., et al.: Perspective on utilizing foundation models for laboratory automation in materials research. Sci. Technol. Adv. Mater. Methods 5(1), 2582379 (2025). <https://doi.org/10.1080/27660400.2025.2582379>
7. Chen, H., et al.: Fast-in-slow: A dual-system foundation model unifying fast manipulation within slow reasoning. arXiv:2506.01953 (2025). <https://arxiv.org/abs/2506.01953>
8. Ahmad, S., Hafeez, M., Zaidi, S.A.R.: Vision-language models on the edge for real-time robotic perception. arXiv:2601.14921 (2026). <https://arxiv.org/html/2601.14921v1>
9. Yang, Z., et al.: EfficientNav: Towards on-device object-goal navigation with navigation map caching and retrieval. arXiv:2510.18546 (2025). <https://arxiv.org/abs/2510.18546>
10. Hughes, M.: AI and robotics: National security implications. Centre for Emerging Technology and Security (CETaS), The Alan Turing Institute (2026). <https://cetas.turing.ac.uk/publications/ai-and-robotics>
11. Hirose, N., Glossop, C., Shah, D., Levine, S.: AsyncVLA: An asynchronous VLA for fast and robust navigation on the edge. arXiv:2602.13476 (2026). <https://arxiv.org/abs/2602.13476>
12. Singh, S.C.E., et al.: AI-driven robotics in agriculture: A review. Premier Journal of Science 15, 100236 (2025). <https://doi.org/10.70389/PJS.100236>
13. Chen, X., Cui, G., Zhang, S., Zhang, X.: Application of navigation path planning and trajectory tracking control methods for agricultural robots. Agriculture 16(1), 64 (2025). <https://www.mdpi.com/2077-0472/16/1/64>
14. Facenda, S., et al.: 3D robotics and LMM for vineyard inspection. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. XLVIII-G-2025, 431–438 (2025). <https://doi.org/10.5194/isprs-archives-XLVIII-G-2025-431-2025>
15. Wang, L., Jin, T., Yang, J., Leonardis, A., Wang, F., Zheng, F.: Agri-LLaVA: Knowledge-infused large multimodal assistant on agricultural pests and diseases. arXiv:2412.02158 (2024). <https://doi.org/10.48550/arXiv.2412.02158>
16. Sun, G., et al.: Special issue editorial: Applications of large language models in Internet of Things. IEEE Internet Things Mag. 8(5) (2025). <https://ieeexplore.ieee.org/iel8/8548628/11197740/11197892.pdf>
17. Khosravi, M., et al.: Optimizing navigation and chemical application in precision agriculture with deep reinforcement learning and conditional action tree. Smart Agric. Technol. 101194 (2025). <https://doi.org/10.1016/j.atech.2025.101194>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

