QUART-Online: Latency-Free Large Multimodal Language Model for Quadruped Robot Learning

Xinyang Tong[1]†, Pengxiang Ding[12]†, Donglin Wang[1]*†, Wenjie Zhang[1], Can Cui[1], Mingyang Sun[12], Yiguo Fan[1]
Han Zhao[12], Hongyin Zhang[12], Yonghao Dang[3], Siteng Huang[12], Shangke Lyu[1]

***Abstract*— This paper addresses the inherent inference latency challenges associated with deploying multimodal large language models (MLLM) in quadruped vision-language-action (QUAR-VLA) tasks. Our investigation reveals that conventional parameter reduction techniques ultimately impair the performance of the language foundation model during the action instruction tuning phase, making them unsuitable for this purpose. We introduce a novel latency-free quadruped MLLM model, dubbed QUART-Online, designed to enhance inference efficiency without degrading the performance of the language foundation model. By incorporating Action Chunk Discretization (ACD), we compress the original action representation space, mapping continuous action values onto a smaller set of discrete representative vectors while preserving critical information. Subsequently, we fine-tune the MLLM to integrate vision, language, and compressed actions into a unified semantic space. Experimental results demonstrate that QUART-Online operates in tandem with the existing MLLM system, achieving real-time inference in sync with the underlying controller frequency, significantly boosting the success rate across various tasks by 65%. Our project page is https://quart-online.github.io.**

I. Introduction

The recent astonishing progress in multimodal large language models (MLLMs) has unveiled their remarkable potential of extracting, aligning, and integrating the representations from complicated language and visual data. These advances have driven the development of a versatile quadruped robot, an embodied agent with vision-language understanding and problem-solving skills, adept at interacting with humans and the physical world to perform complex navigation [1], [2] and whole-body manipulation [3] tasks.

An encouraging preliminary work, QUART [4], has demonstrated the feasibility of adopting MLLMs to control quadruped robots in an end-to-end manner. This not only yields performant robotic policies, but also exhibits some emergent abilities obtained from large models, such as understanding novel commands, generalizing to objects never seen before, and reasoning. Despite these favorable findings, the **high inference latency** is usually an important bottleneck that inhibits the establishment of generalist quadruped robots with advanced MLLMs. Typically, quadruped robotic applications necessitate real-time performance and timely processing for effective action execution and operation in dynamic environments. However, every time MLLMs are activated to obtain a single robotic action involves a computationally intensive inference process. Such inefficiency may prevent real-time integration of perceptual feedback into the decision-making process for actions, thereby leading to the failure of the task. This weakness makes it challenging to deploy MLLMs on robust quadruped robotic systems.

To alleviate this problem, a straightforward approach is to diminish the quantity of parameters of MLLMs. While these methods can reduce inference time, they also compromise the overall performance of the MLLM foundation model, particularly in terms of generalization to unseen scenarios. In Table I, we evaluate QUART with various parameters reduction methods on the QUARD [4] benchmark. While speeding up processing, parameter reduction methods degrade performance, notably in unseen tasks. This suggests a generalization compromise in the foundational capabilities of multimodal models. Striking a balance between efficiency and accuracy in model design requires meticulous consideration, prompting the question of whether enhancing performance is achievable without compromising foundational capabilities.

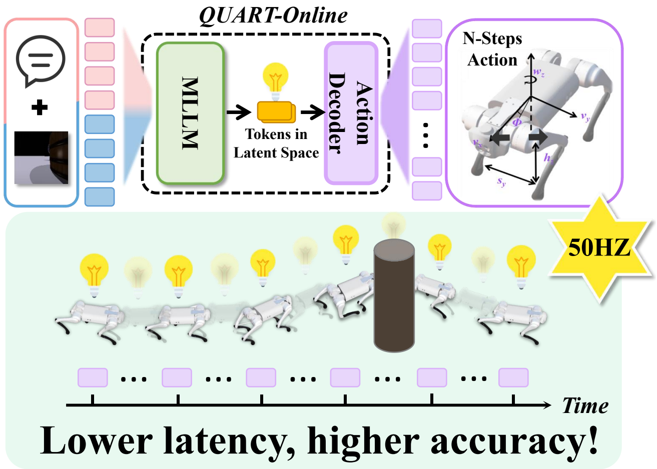

Fig. 1: **Overview of QUART-Online:** With the implementation of action chunk discretization, QUART-Online enhances the existing MLLM system, which was previously operating at a low frequency, enabling more precise actions to be executed in real-time at a frequency of 50Hz.

TABLE I: Success rate and inference speed on the unseen multi-task benchmark with various parameter reduction methods on QUART. Here, "P" signifies the implementation of parameter reduction methods on the QUART.

Methods	QUART	QUART & P	QUART & P
Success Rate	0.74	0.22	0.11
Model Parameters	8B	5.3B	2.7B
Inference Speed	2Hz	3Hz	5Hz

Thus, this paper introduces QUART-Online, a novel latency-free quadruped multimodal large language model, designed to enhance inference efficiency without degrading the performance of the multimodal large language model. Specifically, we first propose a new operation named action chunk discretization (ACD), which maps a sequence of continuous action values onto a smaller set of discrete representative vectors. ACD compresses the original action representation space more effectively

* means corresponding author and † means co-first authors
[1] MiLAB, Westlake University, Hangzhou, 310030, China
[2] Zhejiang University, Hangzhou, 310027, China
[3] Beijing University of Posts and Telecommunications, Beijing, 100876, China
Email of corresponding author: wangdonglin@westlake.edu.cn

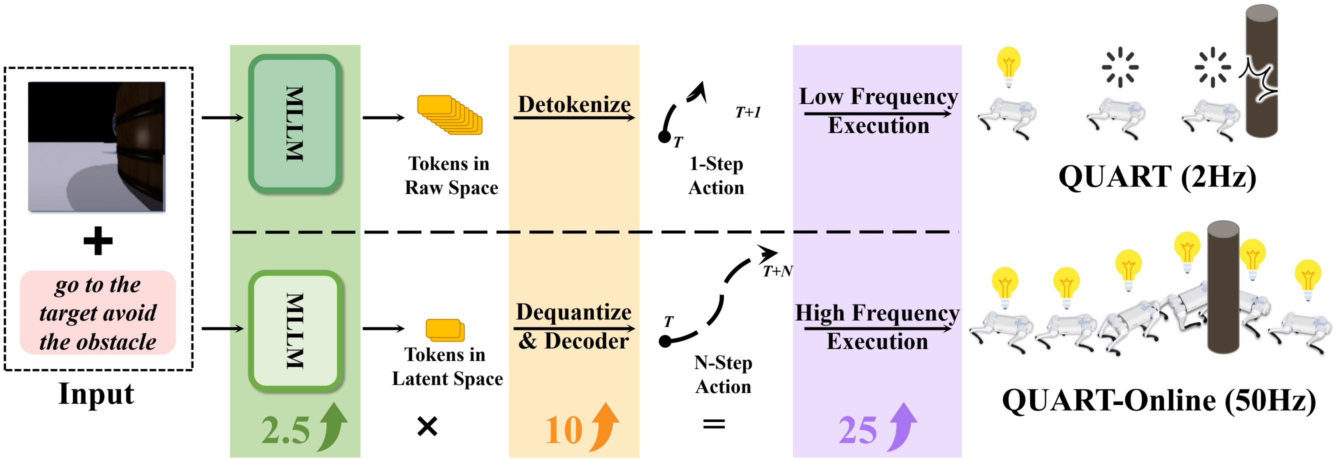

Fig. 2: **Comparison of QUART and QUART-Online.** QUART-Online enhances the inference process by employing two key strategies: 1) it accelerates MLLM inference by generating a reduced number of tokens in the latent space as opposed to the raw space (2.5x); 2) it introduces an action chunk mechanism during the action decoding phase, facilitating higher-frequency inference via multi-step predictions (10x). By integrating these two approaches, QUART-Online successfully increases the inference rate of the original large quadruped robot model, QUART, from 2Hz to 50Hz, enhancing the model's accuracy in rapidly changing scenarios.

while retaining essential information, thereby enhancing the model's output frequency without compromising performance. Subsequently, we fine-tune the MLLM to integrate vision, language, and compressed actions into a unified semantic space. During inference, MLLM first outputs the predicted compressed action tokens, which are then decoded into a continuous trajectory for the robot to execute. By selecting an appropriate action chunk, QUART-Online aligns with the frequency of the underlying controller, facilitating real-time inference. This precise synchronization ensures swift system responses to inputs, resulting in smoother and more coherent action execution, thereby boosting the system's performance and efficiency in task completion.

The performance of QUART-Online is evaluated on various navigation and whole-body manipulation tasks on QUARD [4] benchmark. Extensive experiments have demonstrated that QUART-Online when integrated with existing MLLM systems, achieves real-time inference synchronized with the frequency of the underlying controllers, resulting in a significant 65% improvement in the average task success rate.

II. Related Work

MLLMs for quadruped robot learning. A range of studies [1]–[3], [5]–[21] have explored the use of natural language to instruct quadruped robots in performing tasks. Methods such as SayTap [1] and QuadrupedGPT [21] utilize LLMs as high-level planners to translate commands into individual primitives that low-level controllers then execute. However, these controllers are usually domain-specific small models and lack the semantic understanding and reasoning capabilities that LLMs/MLLMs possess. To fully leverage LLM's astonishing capabilities, QUART [4] harnesses the full potential of LLMs by presenting an end-to-end MLLM that generates robotic actions through finetuning on quadruped data, showcasing emergent abilities like a generalization to novel instructions and objects, as well as reasoning. Despite the potential of projects like QUART in advancing generalist robots, high computational costs often hinder the development of advanced MLLM-equipped quadruped robots, impeding real-time perceptual integration and decision-making. This paper addresses the inference latency challenges in deploying MLLMs for quadruped robot control.

Efficient MLLMs. Significant advancements [22]–[24], [24]–[27] have been made in enhancing the inference efficiency of MLLMs, such as pruning [24], [25], efficient structures [27]. However, the aforementioned methods typically require parameter reduction that are made during the pre-training phase, rather than the downstream fine-tuning stage. Forcibly applying these methods during fine-tuning on robot data would inevitably degrade perceptual comprehension capabilities. Thus, this paper aims to find an approach that achieves efficient inference without altering the model structure and maintains perceptual generalization ability.

Robot Action Representation. Robotic motion encoding primarily falls into two categories: discrete [28]–[31] and continuous [32]–[35]. Given that large models inherently reason within a discrete space, previous end-to-end approaches [28], [29] often employ discrete encoding, mapping all continuous dimensions onto 256 evenly distributed intervals, thereby encapsulating actions as integer values ranging from 0 to 256. This allows the model to align actions with linguistic representations. This mechanism results in inefficient and suboptimal performance, as the model requires multiple inferences to deduce each action dimension for a single frame, adversely affecting inference speed. Additionally, the model's prediction is limited to the immediate next frame, lacking temporal continuity and leading to unstable and imprecise motion generation. To address these issues, we propose action chunk discretization (ACD), which allows MLLMs to synchronize real-time inference with the controller's frequency, significantly boosting the success rate in various quadruped robot tasks.

III. Method

We begin by outlining the quadruped vision-language-action (QUAR-VLA) task and detailing the first VLA model using

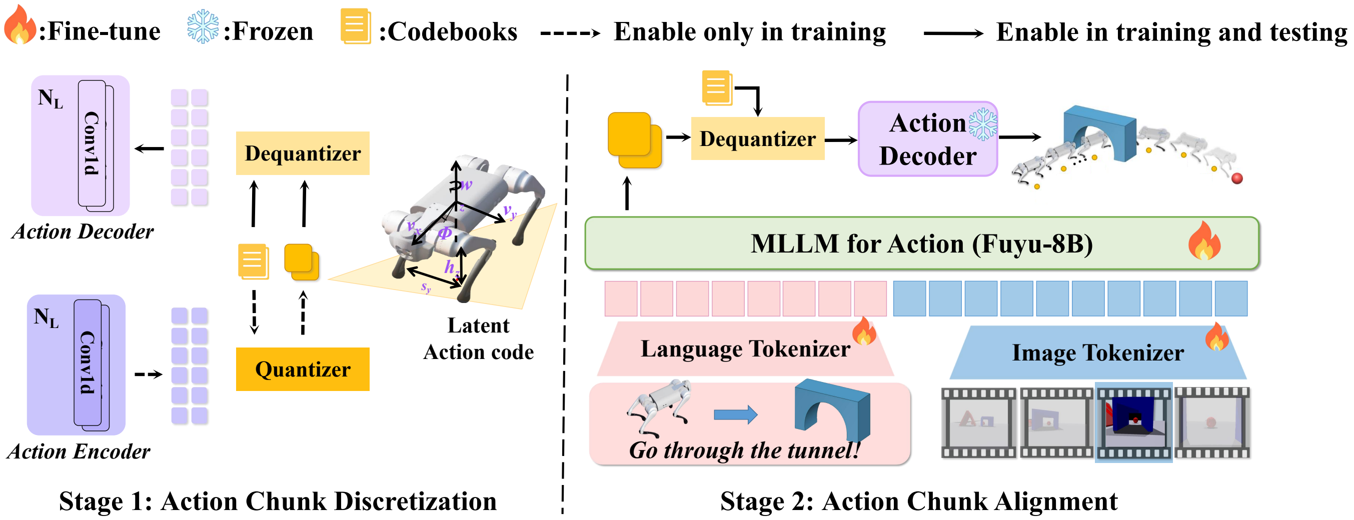

Fig. 3: Overall framework of QUART-Online.

MLLM: quadruped robotic transformer (QUART). Subsequently, we introduce our QUART-Online, a Latency-Free Large Multimodal Language Model for Quadruped Robot Learning from two aspects: **Action Chunk Discretization** and **Action Chunk Alignment**.

A. *Preliminary*

Quadruped Vision-Language-Action (QUAR-VLA) Task. The aim of the QUAR-VLA task is to learn a mapping that interprets RGB images $s \in \mathcal{S}$ alongside task instructions $w \in \mathcal{W}$ to generate future actions through an end-to-end paradigm. This task pays particular attention to the connection between text and visual information and robot dynamics instructions and closely relates perception with robot execution. The mapping is defined as $\mu : \mathcal{S} \times \mathcal{W} \rightarrow \mathcal{A}$, where the action space $\mathcal{A}$ consists of the 11-dimensional high-level commands as well as a terminate signal, i.e.,

$$[v_x, v_y, \omega_z, \theta_1, \theta_2, \theta_3, f, h_z, \phi, s_y, h_z^f, e]. \quad (1)$$

Here, v_x, v_y, and ω_z represent the velocities along the x-axis, y-axis, and z-axis respectively. θ_1, θ_2, and θ_3 indicate the gait pattern, f denotes the frequency, h_z represents the height of the robot, ϕ denotes the pitch angle, s_y corresponds to the foot width, h_z^f represents the foot height, and e indicates the termination signal of the action.

QUAdruped Robotic Transformer (QUART). QUART [4] is the first VLA model that leverage an end-to-end MLLM [36] to directly generate robotic actions via finetuning on collected quadruped data [4]. It takes a single image s and a natural language instruction w as input, which are first converted into concatenated semantic tokens c through tokenizers $\tau(c|s, w)$ and fed into a decoder-only transformer module to obtain predicted action tokens $p(\hat{a_d}|c)$. The whole process could be formulated as follows:

$$\text{QUART}(\hat{a_c}|s,w) = \text{Detokenize}(\hat{a_d})p(\hat{a_d}|c)\tau(c|s,w). \quad (2)$$

where w,s are the input images and language instruction and τ represents the tokenizer and p indicates the vision-language model to output action $\hat{a_d}$. Notably, $\hat{a_d}$ still needs a detokenizer to be converted into continuous representation $\hat{a_c}$ for downstream control.

B. *QUART-Online*

In our pursuit to expedite inference, we initially targeted the most straightforward strategy: model parameter reduction. Conversely, the performance metrics detailed in Table I demonstrate a consistent performance decrement across parameters reduction techniques, with a pronounced decline in efficacy on novel tasks. This trend implies a potential erosion of the intrinsic generalization ability of multimodal models. This is primarily because architectural optimizations during instruction tuning may inadvertently impair the model's foundation ability, consequently diminishing its performance on targeted tasks.

Therefore, we aim to explore a solution that addresses the inference bottleneck without altering the architecture of the large model or its performance; QUART-Online is our solution. As shown in Figure. 2, QUART-Online generates fewer compressed discrete action tokens $\hat{A}_q$ and relies on an action decoder $\mathcal{D}$ to reconstruct $\hat{A}_q$ into continuous action trajectory $\hat{A}_c$:

$$\text{QUART-Online}(\hat{A}_c|s,w) = \mathcal{D}(\hat{A}_c|\hat{A}_q)p(\hat{A}_q|c)\tau(c|s,w), \quad (3)$$

Next, we will illustrate the core two components of QUART-Online: **Action Chunk Discretization** and **Action Chunk Alignment**, where the former maps a sequence of continuous action values onto a smaller set of discrete representative vectors and the latter integrates vision, language, and compressed actions into a unified semantic space when fine-tuning the MLLM.

C. *Action Chunk Discretization*

Let's revisit the discretization method initially employed by QUART, which maps continuous dimensions into 256 distinct bins, representing actions as integers ranging from 0 to 255. This approach encounters two primary issues: redundancy in representation and lack of temporal continuity. Each action frame demands N inference steps, potentially leading to information overlap. Moreover, the action representation fails to maintain temporal consistency and fluidity, causing the generated actions to appear unnatural and disjointed.

To eliminate redundancy, we apply vector quantization to distill raw actions into compressed discrete codes. To ensure temporal coherence, we incorporate it into the learning process of discrete representations, treating multiple consecutive frames as a single chunk. This approach, facilitated by a decoder, allows simultaneous processing of multiple frames, enhancing the fluidity of action generation.

Action Encoder. To efficiently encode a greater number of temporal action steps into a more compact format, we use N_L layers of 1D convolutions on the temporal axis to compactly encode more action steps, ensuring temporal compression and step coherence. This temporal encoder $\mathcal{E}$ maps action sequences into a high-level latent space, allowing for the extraction of rich semantic and temporal details. Formally, we treat the sequence of N consecutive action commands at time T as a set of input action commands $A_c = \{a^i\}_{i=T}^{T+N} \in \mathbb{R}^{N\times 12}$.

$$A_h = \mathcal{E}(A_c), \tag{4}$$

where encoder $\mathcal{E}$ generates latent action tokens $A_h \in \mathbb{R}^{C\times D}$, C denotes the compressed temporal dimension, and D represents the dimension of the latent space.

Quantization. Notably, our method utilizes a multi-layer quantization, where the input data is sequentially quantized by each quantizer, and the residuals from each quantization step are explicitly managed. The learnable codebook $\mathcal{B} \in \mathbb{R}^{N_q\times K\times D}$ consists of N_q layers of codes, where the i-th codebook layer are $\mathcal{B}_i = \{b^k\}_{k=1}^{K}$ including $b^k \in \mathbb{R}^D$. Here K denotes the number of codes in the codebook, and D represents the dimension of the latent space, corresponding to the number of channels obtained from the encoder. The quantization iterates N_q times, culminating in the output of the quantized vector $A_q \in \mathbb{R}^{C\times N_q}$. Specifically, during the i-th quantization layer, the input vector of the layer is compared with the corresponding codebook layer $\mathcal{B}_i$ to determine the closest match, based on distance calculations. The input vector of each layer is the residual from the previous quantization step, representing the discrepancy between the selected embedding vector and the input vector of that layer. By successively quantifying these residuals, the quantization algorithm refines its approximation of the original high-dimensional data, ensuring encoding accuracy while efficiently managing computational complexity.

$$A_q = \text{Quantize}(A_h, \mathcal{B}). \tag{5}$$

Dequantization. The dequantize process is the inverse operation of quantization. Dequantization commences at the apex with layer $\mathcal{B}_{N_q}$ and descends to the foundational layer $\mathcal{B}_1$. Utilizing indices from $\mathcal{A}_q$, it extracts code vectors successively from each layer, aggregating them through a process of accumulation to refine the approximation of the residual incurred at each preceding quantization stage.

$$\hat{A}_h = \text{Dequantize}(\hat{A}_q, \mathcal{B}). \tag{6}$$

Action Decoder. Decoder $\mathcal{D}$, in contrast to the encoder, takes the dequantized hidden action vector $\hat{A}_h$ and restores it to action in the original action space:

$$\hat{A}_c = \mathcal{D}(\hat{A}_h), \tag{7}$$

where $\hat{A}_h \in \mathbb{R}^{N\times 12}$ refers to the action sequence in the original action space that is ultimately reconstructed by the action decoder.

Training Loss. The final quantized vector, which is the result of the residual network reconstruction, is denoted by $\hat{A}_h$. The mean squared error between the original A_h and the reconstructed quantized vector $\hat{A}_h$ is used as the commitment loss, denoted as $\mathcal{L}_{\text{com}}$.

In our approach, we leverage a combination of reconstruction loss and adversarial loss for ACD to achieve high-quality quantization and reconstruction of action tokens.

$$\mathcal{L}_{stage1} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{com}} = \|A_c - \hat{A}_c\|^2 + \|A_h - \hat{A}_h\|^2. \tag{8}$$

D. Action Chunk Alignment

As shown in Fig 3, to empower the MLLM to generate action, we need to jointly optimize and align the compressed actions with language and images within the semantic space.

Training Phase. Upon obtaining a trained vector quantization representation, the target output for the large model transitions from the original discrete values, which span from 0 to 256, to the compressed action tokens. These tokens, in unison with the codebook and the action decoder, facilitate the inference of the final predicted actions. During this phase, the MLLM undergoes fine-tuning, while the action decoder remains fixed. Given that the final predicted action is a continuous representation, the loss function utilized for this stage follows the cross-entropy loss function employed in the QUART training process.

$$\mathcal{L}_{stage2} = -\sum A_q \text{log}(\hat{A}_q). \tag{9}$$

Inference Phase. Employing a precisely aligned and concise action chunk representation, we effectively minimize the inference time, thereby enhancing the smoothness and accuracy of the generated actions. Additionally, by judiciously selecting an appropriate chunk length, we can achieve latent-free control for large quadruped robot models. This is accomplished when the action chunk length l_{ac} multiplied by the frequency of MLLM inference f_m matches the frequency of low-level control f_l. With this well-calibrated chunk length, we can seamlessly implement latency-free control over large quadruped robot models.

$$f_l = l_{ac} \times f_m. \tag{10}$$

IV. Experiments

In this part, we conduct experiments spanning both simulation platforms to address the following inquiries: 1) Can the action chunk discretization (ACD) scheme effectively compress action information? 2) By integrating ACD fine-tuning with MLLM, can QUART-Online maintain model performance while preserving speed? 3) How does the zero-latency inference of QUART-Online enable robots to make better action decisions?

A. Setup

Dataset. Our experimental analysis employs the QUARD dataset [4], a comprehensive, large-scale multi-task dataset known as the Quadruped Robot Dataset (QUARD) [4]. This dataset is rich in content, featuring a variety of tasks including perception, fundamental navigation, and sophisticated capabilities such as whole-body manipulation. Since the QUARD dataset is collected

TABLE II: We use time per action step (frequency) and the success rate as evaluation metrics, conducting 50 experiments for each task. Here U_v and U_l refer to unseen visual elements and unseen language instructions. The numbers in QUART-Online-1, 5, and 10 represent the lengths of the action chunks.

Method	Frequency	Easy		Medium		Hard								Average	
		Distinguish		Go to		Go avoid		Go through		Crawl		Unload			
		U_v	U_l	U_v	U_l	U_v	U_l	U_v	U_l	U_v	U_l	U_v	U_l	U_v	U_l
VLA (CLIP)	43HZ	0.40	0.56	0.66	0.27	0.53	0.22	0.31	0.06	0.00	0.2	0.01	0.00	0.32	0.22
VLA (VC-1)	40HZ	0.76	0.63	0.37	0.58	0.34	0.67	0.15	0.16	0.00	0.03	0.00	0.00	0.27	0.35
QUART	2HZ	0.80	0.89	0.59	0.76	0.2	0.41	0.23	0.40	0.32	0.46	0.08	0.18	0.37	0.52
QUART-Online-1	5HZ	0.48	0.75	0.25	0.76	0.05	0.33	0.33	0.68	0.56	0.5	0.00	**0.42**	0.28	0.57
QUART-Online-5	25HZ	0.82	0.99	0.61	0.90	0.39	0.54	0.26	0.42	0.48	0.59	0.06	0.28	0.47	0.62
QUART-Online-10	50HZ	**0.90**	**0.99**	**0.89**	**1.00**	**0.58**	**0.80**	**0.76**	**0.94**	**0.79**	**0.78**	**0.18**	0.24	**0.68**	**0.79**

TABLE III: Reconstruction performance of action chunk discretization. Here l_{ac} means the length of the action chunk.

l_{ac}	MAE(↓)	AKI(↓)	PSNR(↑)	UQI(↑)
1	0.028	0.037	25.57	0.9993
5	0.013	0.0009	31.89	0.9998
10	**0.012**	**0.0008**	**32.11**	**0.9999**

within NVIDIA's Isaac Gym [37], we chose to use the same simulator as our experimental evaluation environment.

Evaluation. 1) Environment: We replicate the QUARD dataset configuration for our evaluation, employing a policy model that outputs high-level commands, which are then translated into robot joint movements by a low-level controller [38]. 2) Tasks: Our analysis extends beyond performance across multiple task scenarios to encompass the model's capability to generalize to novel contexts featuring unseen visual components (U_v) and innovative language instructions (U_l). For visual components, we examined items with different textures, colors, and shapes within the same category, as well as those with shape and texture variations not present in current datasets. For language commands, we tested adaptability by using synonymous but differently phrased instructions, such as "navigate to target" instead of "go to the object." 3) Metrics: We evaluate model accuracy and real-time performance based on success rate and execution frequency. The action compression precision during the ACD phase is quantified using metrics such as mean absolute error (MAE), average kurtosis index (AKI), peak signal-to-noise ratio (PSNR), and universal image quality index (UQI).

Baselines. For a thorough comparative analysis, we follow the experimental setup of [4], [29], incorporating the QUART model [4] as a paradigm of methods that leverags large language models. Furthermore, we have integrated foundational vision models [39], [40] with linguistic models to construct distinct vision-language-action (VLA) baselines following the RT-1 [28] paradigm: VLA (CLIP) and VLA (VC-1).

Implementation Details. In the ACD process, we employ a 1D convolutional Encoder-Decoder with a kernel size of 4 and a stride of 1. The quantization network features a codebook embedding layer with 512 dimensions and 512 quantizers ($N_q = 2$). We predict model reconstruction at lookahead steps of N = 5 and N = 10, with 3 convolutional layers at N = 5. The training utilizes AdamW with a learning rate of 3e-4, optimizing the mean squared error loss. For the multimodal large model, we adopt the lightweight Fuyu-8b as our pre-trained base. Training parameters include a learning rate of 2e-5, AdamW optimizer, 10 epochs, 100K gradient steps, and a batch size of 256. We have addressed previous bugs in the QUART evaluation environment, ensuring all new baselines are reproducible in the updated evaluation setup.

B. Fidelity of action chunk discretization (ACD).

Given QUART-Online's performance reliance on the quality of ACD, we first assessed the reconstruction accuracy of trajectory discretization. In this study, we allocated 85% of the action data for training and reserved 15% for validation. During the validation phase, we tested three different action chunk lengths (1, 5, and 10) and computed the reconstruction metrics shown in Table III to evaluate the fidelity of the trajectory discretization process. The results indicate that as the action chunk length increases, the reconstruction accuracy significantly improves. This enhancement is primarily due to the introduction of additional temporal information, which offers more cues for the reconstruction process. Furthermore, the results in Table II demonstrate that as the action chunk length increases, QUART-Online's overall performance also shows significant enhancement. This suggests that the increased temporal information improves not only the quality of action reconstruction but also the model's decision-making accuracy. The following section will elaborate on this in detail.

C. Comparison with SOTA methods

We evaluate the model's performance from two dimensions: its success rate to different scenarios including generalization to previously unseen vision components (U_v) and language instructions (U_l), as well as its real-time responsiveness. The whole results are shown in Table II.

QUART-Online versus QUART. Compared to QUART, QUART-Online has achieved significant improvements in success rates across various tasks. These results demonstrate that QUART-Online significantly enhances multi-task and generalization capabilities compared to QUART, particularly in managing unseen objects and instructions. This improvement arises from the unified optimization of actions and perceptual data within a discrete space. Unlike QUART, which compromises semantic distribution by representing actions as discrete numerical tokens aligned with language signals, QUART-Online discretizes actions to extract compressed tokens that convey inherent semantic meaning. This

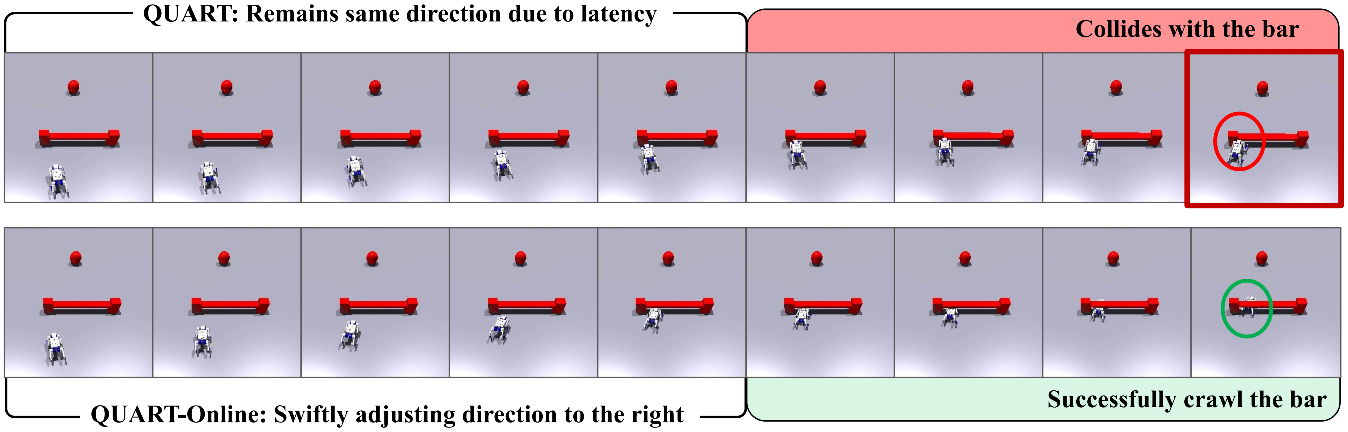
QUART: Remains same direction due to latency
Collides with the bar
QUART-Online: Swiftly adjusting direction to the right
Successfully crawl the bar

Fig. 4: The top half of the comparison highlights the QUART method's latency-induced collision with the red bar (red highlight). The bottom half displays the QUART-Online method's agile response, successfully avoiding the obstacle (green highlight).

approach allows for joint optimization without disrupting the model's existing distribution, leading to better performance in novel scenarios. Additionally, longer action chunk durations are associated with improved outcomes, suggesting that the encoding of extended time sequences fosters reasoning abilities linked to the current environment, enabling more effective task execution in unfamiliar contexts. Furthermore, QUART-Online achieves a significant increase in inference frequency, improving from 2Hz to 50Hz. This enhancement enables QUART-Online to operate seamlessly with the existing MLLM system, facilitating real-time inference that aligns closely with the underlying controller's frequency.

QUART-Online versus other baseline models. Compared to other baseline models, QUART-Online demonstrates superior performance across all tasks while maintaining a similar inference rate. This reinforces the effectiveness of our approach, which combines the benefits of large models with the real-time capabilities of lightweight models. Table II illustrates that QUART-Online achieves high success rates, particularly in the "Go avoid" and "Crawl" tasks, outperforming VLA (CLIP) and VLA (VC-1). This combination of high success rates and increased inference frequency enhances the applicability of large models in dynamic robotic tasks.

D. *Latency-Free Performance Analysis*

This part explores strategies for coping with delays in dynamic environments by comparing the performance of QUART and QUART-Online. Figure 4 illustrates the different responses of the two algorithms when facing the same obstacle (the red horizontal bar). In the case of QUART, the system failed to adjust its direction in time due to delays. As the red ball moved, the robot maintained its original direction, ultimately leading to a collision with the bar (marked by the red box). This result indicates that QUART reacts slowly to dynamic targets and cannot effectively avoid obstacles, highlighting its limitations in real-time applications. In contrast, QUART-Online demonstrated a more flexible response strategy. This algorithm was able to quickly adjust its direction and timely avoid the bar, successfully passing through (marked by the green box). This shows that QUART-Online has better real-time feedback capabilities and can make more effective decisions in dynamic environments. The experimental results indicate that QUART-Online outperforms QUART under delay conditions, handling dynamic obstacles more effectively.

V. Discussion and Limitations

QUART-Online operates in tandem with the existing MLLM system, achieving real-time inference in sync with the underlying controller frequency, significantly boosting the success rate across various tasks. However, there also exist several limitations: 1) The current actions still do not reach the level of joint angles and still require low-level control to obtain joint angle actions. Therefore, exploring how to directly reach the joint angle layer with a large model in the future is a worthwhile area of research. 2) Our rapid-inferring model has yet to be tested in more complex locomotion scenarios. Ensuring that our efficient inference model can adapt to various complex terrains and further tapping into its potential will be the focus of our next phase of work.

VI. Conclusion

This paper addresses the challenge of inference latency in MLLMs for controlling quadruped robots. It critiques traditional approaches to parameter reduction, which often lead to a decline in model performance during the downstream tuning phase. In response, the paper introduces QUART-Online, an innovative latency-free method designed to enhance inference efficiency without sacrificing the integrity of the large language model. QUART-Online employs ACD to effectively compress the action space while preserving essential information for precise action mapping. By fine-tuning with perception data within a unified semantic space, QUART-Online is capable of operating in concert with existing MLLM systems, achieving real-time and accurate inference that is synchronized with the underlying controller's frequency. QUART-Online significantly steps forward in the development of efficient and responsive robotic control solutions for integration with MLLM systems.

References

[1] Y. Tang, W. Yu, J. Tan, H. Zen, A. Faust, and T. Harada, "Saytap: Language to quadrupedal locomotion," *arXiv preprint arXiv:2306.07580*, 2023.

[2] S. Kareer, N. Yokoyama, D. Batra, S. Ha, and J. Truong, "Vinl: Visual navigation and locomotion over obstacles," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2018–2024, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:253117178

[3] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 807–11 814, 2022.

[4] P. Ding, H. Zhao, Z. Wang, Z. Wei, S. Lyu, and D. Wang, "Quar-vla: Vision-language-action model for quadruped robots," *arXiv preprint arXiv:2312.14457*, 2023.

[5] K. Caluwaerts, A. Iscen, J. C. Kew, W. Yu, T. Zhang, D. Freeman, K.-H. Lee, L. Lee, S. Saliceti, V. Zhuang *et al.*, "Barkour: Benchmarking animal-level agility with quadruped robots," *arXiv preprint arXiv:2305.14654*, 2023.

[6] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," *ArXiv*, vol. abs/2107.03996, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235765481

[7] Y. Wang, R. Sagawa, and Y. Yoshiyasu, "Learning advanced locomotion for quadrupedal robots: A distributed multi-agent reinforcement learning framework with riemannian motion policies," *Robotics*, vol. 13, no. 6, 2024. [Online]. Available: https://www.mdpi.com/2218-6581/13/6/86

[8] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto, "The surprising effectiveness of representation learning for visual imitation," 2021.

[9] G. Schiavi, P. Wulkop, G. Rizzi, L. Ott, R. Siegwart, and J. J. Chung, "Learning agent-aware affordances for closed-loop interaction with articulated objects," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5916–5922.

[10] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak, "Affordances from human videos as a versatile representation for robotics," 2023.

[11] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019. [Online]. Available: https://arxiv.org/abs/1810.10191

[12] S. Belkhale, Y. Cui, and D. Sadigh, "Hydra: Hybrid robot actions for imitation learning," in *Proceedings of the 7th Conference on Robot Learning (CoRL)*, 2023.

[13] S. Haldar, V. Mathur, D. Yarats, and L. Pinto, "Watch and match: Supercharging imitation with regularized optimal transport," *CoRL*, 2022.

[14] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abk2822

[15] Z. He, K. Lei, Y. Ze, K. Sreenath, Z. Li, and H. Xu, "Learning visual quadrupedal loco-manipulation from demonstrations," 2024. [Online]. Available: https://arxiv.org/abs/2403.20328

[16] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, and S. Levine, "Genloco: Generalized locomotion controllers for quadrupedal robots," 2022. [Online]. Available: https://arxiv.org/abs/2209.05309

[17] S. Jeon, M. Jung, S. Choi, B. Kim, and J. Hwangbo, "Learning whole-body manipulation for quadrupedal robot," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 699–706, 2024.

[18] Y. Ouyang, J. Li, Y. Li, Z. Li, C. Yu, K. Sreenath, and Y. Wu, "Long-horizon locomotion and manipulation on a quadrupedal robot with large language models," 2024. [Online]. Available: https://arxiv.org/abs/2404.05291

[19] A. Jaafar, S. S. Raman, Y. Wei, S. E. Juliani, A. Wernerfelt, I. Idrees, J. X. Liu, and S. Tellex, "Lanmp: A multifaceted mobile manipulation benchmark for robots," in *RSS 2024 Workshop: Data Generation for Robotics*.

[20] J. Wu, Y. Xue, and C. Qi, "Learning multiple gaits within latent space for quadruped robots," 2023. [Online]. Available: https://arxiv.org/abs/2308.03014

[21] Y. Wang, Y. Mei, S. Zheng, and Q. Jin, "Quadrupedgpt: Towards a versatile quadruped agent in open-ended worlds," *arXiv preprint arXiv:2406.16578*, 2024.

[22] Z. Wan, X. Wang, C. Liu, S. Alam, Y. Zheng, Z. Qu, S. Yan, Y. Zhu, Q. Zhang, M. Chowdhury *et al.*, "Efficient large language models: A survey," *arXiv preprint arXiv:2312.03863*, vol. 1, 2023.

[23] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang, "Dynamic neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7436–7456, 2021.

[24] M. Hinck, M. L. Olson, D. Cobbley, S.-Y. Tseng, and V. Lal, "Llava-gemma: Accelerating multimodal foundation models with a compact language model," *arXiv preprint arXiv:2404.01331*, 2024.

[25] B. Lin, Z. Tang, Y. Ye, J. Cui, B. Zhu, P. Jin, J. Zhang, M. Ning, and L. Yuan, "Moe-llava: Mixture of experts for large vision-language models," *arXiv preprint arXiv:2401.15947*, 2024.

[26] H. Zhao, M. Zhang, W. Zhao, P. Ding, S. Huang, and D. Wang, "Cobra: Extending mamba to multi-modal large language model for efficient inference," *arXiv preprint arXiv:2403.14520*, 2024.

[27] J. Mercat, I. Vasiljevic, S. Keh, K. Arora, A. Dave, A. Gaidon, and T. Kollar, "Linearizing large language models," *arXiv preprint arXiv:2405.06640*, 2024.

[28] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "Rt-1: Robotics transformer for real-world control at scale," 2023.

[29] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," 2023.

[30] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh, "Rt-h: Action hierarchies using language," 2024. [Online]. Available: https://arxiv.org/abs/2403.01823

[31] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," 2024. [Online]. Available: https://arxiv.org/abs/2406.09246

[32] H. Wu, Y. Jing, C. Cheang, G. Chen, J. Xu, X. Li, M. Liu, H. Li, and T. Kong, "Unleashing large-scale video generative pre-training for visual robot manipulation," 2023. [Online]. Available: https://arxiv.org/abs/2312.13139

[33] X. Li, M. Liu, H. Zhang, C. Yu, J. Xu, H. Wu, C. Cheang, Y. Jing, W. Zhang, H. Liu, H. Li, and T. Kong, "Vision-language foundation models as effective robot imitators," 2024. [Online]. Available: https://arxiv.org/abs/2311.01378

[34] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3d diffuser actor: Policy diffusion with 3d scene representations," 2024. [Online]. Available: https://arxiv.org/abs/2402.10885

[35] A. Szot, B. Mazoure, H. Agrawal, D. Hjelm, Z. Kira, and A. Toshev, "Grounding multimodal large language models in actions," 2024. [Online]. Available: https://arxiv.org/abs/2406.07904

[36] R. Bavishi, E. Elsen, C. Hawthorne, M. Nye, A. Odena, A. Somani, and S. Taşırlar, "Introducing our multimodal models," 2023. [Online]. Available: https://www.adept.ai/blog/fuyu-8b

[37] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[38] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 22–31.

[39] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021.

[40] A. Majumdar, K. Yadav, S. Arnaud, Y. J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, P. Abbeel, J. Malik, D. Batra, Y. Lin, O. Maksymets, A. Rajeswaran, and F. Meier, "Where are we in the search for an artificial visual cortex for embodied intelligence?" 2023.