Extending on Speculative Decoding for Vision-Language-Action Models in Robotics

Stanford CS229 Project, Dec 5 2025

Hadil Owda
Department of Computer Science
Stanford University
`hadilo12@stanford.edu`

Josh Bowden
Department of Computer Science
Stanford University
`jjosh@stanford.edu`

Darrow Hartman
Department of Computer Science
Stanford University
`darrowrh@stanford.edu`

Abstract

Vision-Language-Action (VLA) models demonstrate impressive generalization in robotic manipulation but suffer from prohibitive inference latency due to autoregressive generation. While speculative decoding has been proposed to alleviate this bottleneck as it has in LLMs, early existing approaches rely on relaxed acceptance criteria to achieve speedups, sacrificing the identical distribution guarantee. In this work, we investigate whether Mamba, a State Space Model with linear complexity, can serve as a superior draft model to standard Transformers for *strict* speculative decoding. We integrated a Mamba-based draft model with OpenVLA and evaluated performance on the LIBERO-Goal benchmark. Contrary to our hypothesis, we found that the Mamba draft model (23.29sec/episode) performed on par with a replicated Llama draft baseline (22.61sec/episode) and the autoregressive baseline (23.38sec/episode), yielding no significant speedup. We also found that SpecVLA (EMNLP '25) failed to replicate their 1.09x speedup with speculative decoding. These results highlight the difficulty of accurately modeling information-dense robot action distributions for speculation.

1 Introduction

Vision-language-action (VLA) models have emerged as a powerful approach for robotic manipulation, enabling robots to generate low-level control actions directly from visual observations and natural language instructions. By integrating visual perception, language understanding, and motor control in a unified framework, VLA models like OpenVLA demonstrate impressive generalization across diverse manipulation tasks. However, their autoregressive token generation process creates a critical inference bottleneck: generating action commands requires sequentially sampling 7 action tokens through autoregressive decoding, where each token prediction depends on all previously generated tokens. This sequential dependency severely limits deployment in real-time robotic control loops where sub-second response times are essential.

Speculative decoding (SD) addresses this bottleneck by employing a smaller, faster draft model to speculatively generate candidate token sequences that are then verified in parallel by the target model. When the draft model's predictions align with the target model, multiple tokens can be accepted in a single verification step, achieving significant speedups without changing the output distribution. SpecVLA adapts this technique to vision-language-action models, demonstrating substantial inference speedups on robotic manipulation benchmarks. However, to achieve these gains, SpecVLA requires relaxed acceptance thresholds, which compromises the mathematical guarantee that the draft and target distributions remain identical.

Input and Output: The input to our system consists of robot visual observations (RGB images) and natural language task instructions (e.g., "pick up the red block"). Our draft model processes hidden state representations from the OpenVLA vision encoder (tensors of shape $[\text{sequence\_length}, 4096]$) along with token embeddings, and outputs predicted hidden states that are mapped to action tokens, which represent a 7-dimensional action space representing end-effector control (delta position, rotation and gripper) for a robot arm.

Our Approach: This work investigates whether replacing SpecVLA's Llama-based draft model with Mamba-based state-space models can improve draft accuracy and enable strict speculative decoding while maintaining or exceeding SpecVLA's speedups. We hypothesize that Mamba's linear-time sequential modeling and selective state-space mechanism may be better suited for predicting temporally-coherent action sequences than attention-based architectures. We implement and evaluate Mamba draft models against the SpecVLA baseline on the LIBERO manipulation benchmark.

2 Related Work

Prior work on VLA models can be grouped into three methodological categories: (1) large VLM-based VLA systems, (2) end-to-end policy models integrating state-space or transformer architectures, and (3) inference-time acceleration and speculative decoding for robotic action generation. Below, we contextualize each line of work, highlight their strengths and limitations, and discuss how they differ from our approach.

A growing body of research explores VLA systems built on large pretrained vision–language models. Shao et al. provides a comprehensive survey of large VLM-based VLA models, organizing the field into monolithic architectures, which jointly process perception and action, and hierarchical pipelines, which introduce intermediate abstractions such as subgoals, keypoints, or programmatic plans (Shao et al. (2025)). These works leverage techniques including reinforcement learning, imitation learning from human videos, world-model reasoning, and training-free optimization. The survey identifies several open challenges—long-horizon memory, 4D perception, efficient adaptation, and multi-agent coordination—highlighting that while VLMs provide strong generalization priors, their grounding in real-world action spaces remains difficult. This line of work establishes the motivation for building scalable VLA systems but does not directly address inference efficiency, which is critical for real-time manipulation.

OpenVLA represents a parallel line of work focused on building a scalable and unified vision–language–action foundation model for real-world robotics (Kim et al. (2024)). Trained on 970k robot-interaction sequences spanning 581 skills and 90 hours of diverse data, OpenVLA demonstrates strong zero-shot and few-shot generalization across embodiments and tasks. Its architecture extends large-scale VLMs with an action head capable of autoregressively predicting continuous robot actions, enabling cohesive multimodal reasoning and execution. Crucially, OpenVLA is designed as a high-capacity verifier model whose semantic grounding and broad skill coverage make it suitable for downstream control pipelines—including speculative decoding frameworks such as Spec-VLA. However, the model's scale (7B parameters) imposes nontrivial inference latency, motivating the need for lightweight draft models capable of accelerating action generation without sacrificing OpenVLA's accuracy.

Recent models aim to integrate perception, language grounding, and control in compact architectures that can be trained with limited robot data. RoboMamba combines a CLIP-based visual encoder with the Mamba state-space model, enabling linear-time sequence processing and improved temporal reasoning (Liu et al. (2024)). Through alignment of visual tokens with language embeddings, RoboMamba acquires general commonsense capabilities, and with a lightweight policy head (only 0.1% of parameters), it predicts SE(3) end-effector poses with strong task performance and a $3\times$ runtime speedup compared to prior transformer-based VLA models. These results suggest that efficient state-space models provide a promising alternative to heavy autoregressive language models.

As VLA models grow in scale, reducing inference latency becomes increasingly important for closed-loop robotics. Speculative decoding, originally developed for accelerating large language models, proposes a two-model pipeline in which a lightweight draft model generates candidate tokens that are subsequently verified by a larger target model (Leviathan et al. (2023)). Spec-VLA extends this idea to robotic action spaces (Wang et al. (2025)). By combining a draft VLA model with OpenVLA as the verifier and introducing a distance-based relaxed acceptance rule, Spec-VLA widens the set of candidate tokens that count as valid, reflecting the inherent continuity of robot motion spaces. This modification increases the acceptance rate by 25–44% and yields a $1.22\times$–$1.42\times$ speedup without degrading task success. While speculative decoding is widely used in language modeling, Spec-VLA demonstrates that adapting it to continuous or quasi-continuous action representations requires principled modifications to the acceptance criterion.

Collectively, prior work demonstrates that (1) large VLMs provide strong generalization and semantic grounding; (2) efficient sequence models such as Mamba can significantly reduce computational load; and (3) inference-time acceleration techniques such as speculative decoding are essential for real-time robotic control. However, these approaches either focus on representational power (VLM-based systems and OpenVLA), architectural efficiency (RoboMamba), or inference-time speed (Spec-VLA), but rarely all three. Our work builds on these insights by using Mamba as a draft model to examine how it can speed up the speculative decoding pipeline. We aim to show that combining efficient architectural design with tailored speculative decoding can further close the gap between high-capacity multimodal reasoning and real-time robotic manipulation.

3 Dataset and Features

We create our dataset by running OpenVLA on the LIBERO-Goal benchmark (Liu et al., 2023). The data consists of observation-language-action triplets, with 256x256 images, the language instruction of the task, and an action token which represents one of 256 7-dimensional action space representing end-effector control (xyz delta position, xyz delta rotation and gripper open/close) for a robot arm.

The dataset also includes with every triplet the hidden state representations from the OpenVLA vision encoder (tensors of shape [sequence length, 4096]) along with token embeddings for the observations and prompts.

The dataset is 170GB and contains 26,000 observation-language-action triplets after processing out no-op actions (originally 52,000 pairs). See the below figure for example observations. An example of a task prompt is "Put the bowl on the stove".

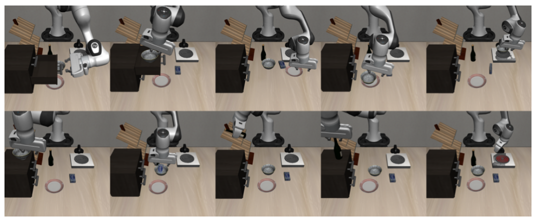

Figure 1: LIBERO-Goal initial states, Liu et. al. 2023

4 Methods

Speculative decoding is a technique for accelerating autoregressive inference in large language models. The key idea is to use a smaller, faster draft model to generate multiple candidate tokens in parallel, which are then verified by the larger target model in a single forward pass. If the draft model's predictions match what the target model would have generated, those tokens are accepted and the generation process skips ahead multiple steps. If predictions diverge, the target model's output is used instead and generation continues normally. This approach maintains an identical output distribution to standard autoregressive generation while achieving 2-3x speedups in practice, particularly effective on text where many tokens (like filler words or common phrases) are relatively easy to predict.

We make use of SpecVLA's code implementing speculative decoding and EAGLE top-k sampling from the draft model (Li et al., 2024)so that we can focus on training and evaluating draft models.

We aim to improve the application of speculative decoding (SD) to vision-language-action models in robotics. SpecVLA adapted this approach by using a Llama decoder layer as the draft model for OpenVLA. However, they found that 50% of the time, the draft model's predictions were completely incorrect, achieving only 1.1x speedup. This poor performance led them to introduce a relaxed acceptance criterion that allows approximately correct predictions.

Our approach is to develop draft models with architectures better suited to predicting robot actions, improving both prediction accuracy and speedup without requiring relaxed acceptance criteria.

Our main metric is seconds per episode, which describes how quickly we can run inference. A fast and accurate draft model will result in fewer calls to the large model, which leads to faster speed.

5 Experiments / Results / Discussion

5.1 Experiment 1: Mamba draft model vs replication of SpecVLA autoregressive and draft model approaches

We train a 2-layer Mamba (Gu and Dao, 2024) model with a model dimension of 4096, state dimension of 16, and expansion factor of 2. The model is intialized with a frozen embedding layer of the base OpenVLA model for compatibility wit hidden states. We train for 200 epochs with a batch size of 4, learning rate of 5e-5, and 2000 warmup steps using AdamW optimizer and bf16 precision for larger range than fp16 to avoid RNN overflow. The objective function is a weighted combination of a regression loss (SmoothL1Loss) on the hidden states and a classification loss (LogSoftmax/CrossEntropy) on the output logits. Training took 24 hours on a H200 GPU.

We also replicated the SpecVLA Llama draft model training, with the same details as above, taking 24 hours on an H100 GPU. The main difference is fp16 precision.

Evaluating on the 10 LIBERO-Goal tasks, with 10 episodes with different initial layouts per task, we found that strict speculative decoding with Mamba draft model resulted in 23.29 seconds/episode on average, with a standard deviation of 12.49 seconds. This is in the same ballpark as the Llama draft model (22.61 sec/ep, SD 12.15sec) and the autoregressive approach (23.38 sec/ep, SD 13.01sec). Evaluation took about 1 hour on an H100 GPU for each modeling approach.

Table 1: Comparison of time per episode, standard deviation, and accuracy across different decoding approaches.

Model / Approach	**Avg. Time (s)**	**Std. Dev. (s)**	**Accuracy (%)**
Autoregressive	23.38	13.01	69.00
Speculative (Llama Draft)	22.61	12.15	71.00
Speculative (Mamba Draft)	23.29	12.49	71.00

Based on these results, we failed to replicate the findings from SpecVLA of a 1.08-1.15x speedup (1.09x on LIBERO-Goal), instead finding a 1.03x speedup with their draft model. We also find no speedup with a Mamba draft model. Note that the accuracy differences are only due to random sampling, as strict speculative decoding rejects inaccurate tokens from draft models. Considering this random variation, there is no robust difference between the average time of all three approaches.

We don't believe that we overfit on the training set because accuracy remains mid-level and LIBERO-Goal has built-in data augmentation (random initial positions for objects), and we tested on different initial positions than we trained on.

5.2 Experiment 2: Integrating RoboMamba as a Draft Model for SpecVLA

We investigated replacing SpecVLA's EAGLE draft model with RoboMamba, a vision-conditioned Mamba architecture that generates continuous action predictions in a single forward pass. RoboMamba consists of a CLIP vision encoder, an MLP connector aligning visual and language representations, a Mamba SSM backbone, and lightweight policy heads that regress end-effector pose parameters Liu et al. (2024).

Integrating RoboMamba into SpecVLA required resolving fundamental mismatches in action representation and model structure. SpecVLA's verifier, OpenVLA, operates in a discrete action space, encoding each of seven action dimensions as a token quantized into 256 bins, whereas RoboMamba outputs continuous values in an eight-dimensional regression space. To bridge this gap, we introduced an 8D-to-7D projection and a discretization pipeline that maps continuous predictions to OpenVLA's action tokens.

The two models also differ substantially in hidden-state dimensionality: RoboMamba's Mamba-2.8B backbone produces 2560-dimensional states, while OpenVLA's LLaMA-7B verifier expects 4096-dimensional draft representations. We therefore added a learned linear projection to align RoboMamba hidden states with the verifier's embedding space.

Overall, adopting RoboMamba as a draft model required explicit alignment modules for both actions and hidden states, revised verification logic, and additional training infrastructure.

On the systems side, we extended the `EaModel` class to support RoboMamba as an alternative draft model by adding configuration flags, checkpoint handling, and routing logic. We implemented a `RoboMambaDraftModel` wrapper that loads the vision–language backbone, constructs 2D and 6D action heads, applies hidden-state projection, converts 8D actions to OpenVLA's 7D format, and discretizes them into verifier-compatible tokens. Action conversion is handled by an `ActionDiscretizer`, which normalizes continuous outputs and maps them to integer bins aligned with OpenVLA's vocabulary.

For verification, we introduced a linear procedure, `verify_block_draft`, which performs a single forward pass through OpenVLA on the prompt and draft sequence and filters draft tokens using relaxed acceptance criteria. We additionally added flexible weight-loading support for multiple projection and action-head formats, and implemented two training pipelines: one that trains RoboMamba's action heads on LIBERO with a frozen backbone, and another that jointly trains the hidden-state and action projections to better align RoboMamba outputs with OpenVLA.

Despite these efforts, the resulting system did not produce a functional draft model. We attribute this failure to insufficient alignment between RoboMamba's continuous, low-dimensional latent representations and SpecVLA's discrete-token verifier interface. Effective block drafting requires jointly trained action heads, a learned hidden-state projection, and exact consistency between discretization bins and verifier vocabulary indices. When any of these components are missing or mismatched—such as using a text-only RoboMamba checkpoint without trained manipulation heads, omitting the projection layer, or misaligning action-token ranges—the draft outputs fail to condition the OpenVLA verifier. In practice, this leads to high rejection rates or invalid verification behavior, preventing speculative speedups.

6 Conclusion / Future Work

Our Mamba draft model had a similar speedup to their Llama draft model. However, we disappointingly failed to replicate SpecVLA's original result of a 1.09x speedup on LIBERO-Goal with the same configs and compute, instead getting a speedup of 1.03x. This aligns with one issue on their Github, which is dissappointing for a paper accepted to a prominent conference (EMNLP 25). Since robot actions are inherently more information-dense than language, it seems that it remains hard to make speculative decoding effective on robotics.

With more time, we would analyze the acceptance length of the Mamba draft model to gain more insight into how it balanced accuracy and speed. We would also look more into per-task speed, because all three approaches had a very high standard deviation (half of the mean). We also failed to integrate the roboMamba model into SpecVLA as a draft model. If we had more time we could experiment with better pipelining between the draft model and the target model. Moreover, a longer training run would have been beneficial to reduce the loss in the retrained roboMamba draft model.

Speculative decoding is complex and tricky to debug and analyze, so in the future we would aim to make a simpler testing harness that can isolate the draft models in an easier way. We would want a more traditional test dataset where the draft model could be tested for speed and accuracy, instead of needing to integrate into a very complicated codebase.

7 Contributions

Josh - replicated training of SpecVLA Llama draft model, wrote training script and configs for Mamba training, ran training on H100/H200s, adapted evaluation script for Mamba model, evaluated Mamba draft model versus SpecVLA, worked on report

Darrow - wrote training and test scripts for integrating roboMamba model into SpecVLA, researched related work in VLAs, replicated SpecVLA and roboMamba repositories, contributed to final report

Hadil - attempted designeding and implemented Mamba-based draft model architecture variation. Set up development environment in Google Colab. Contributed to literature review and final report.

References

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. 2024. Openvla: An open-source vision-language-action model.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024. Eagle-2: Faster inference of language models with dynamic draft trees.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. 2023. Libero: Benchmarking knowledge transfer for lifelong robot learning.

Jiaming Liu, Mengzhen Liu, Zhenyu Wang, Pengju An, Xiaoqi Li, Kaichen Zhou, Senqiao Yang, Renrui Zhang, Yandong Guo, and Shanghang Zhang. 2024. Robomamba: Efficient vision-language-action model for robotic reasoning and manipulation.

Rui Shao, Wei Li, Lingsen Zhang, Renshan Zhang, Zhiyang Liu, Ran Chen, and Liqiang Nie. 2025. Large vlm-based vision-language-action models for robotic manipulation: A survey.

Songsheng Wang, Rucheng Yu, Zhihang Yuan, Chao Yu, Feng Gao, Yu Wang, and Derek F. Wong. 2025. Spec-vla: Speculative decoding for vision-language-action models with relaxed acceptance.