Learning Feasibility from Failure Data in Vision–Language–Action Models

Jeongeun Park[1], **Jihwan Yoon**[1], **Byungwoo Jeon**[2], **Juhan Park**[1], **Namhoon Cho**[3], **Kyungjae Lee**[1], **Sangdoo Yun**[4], **Sungjoon Choi**[1]
[1] Korea University, [2] Korea Advanced Institute of Science & Technology,
[3] Seoul National University, [4] Naver AI Lab

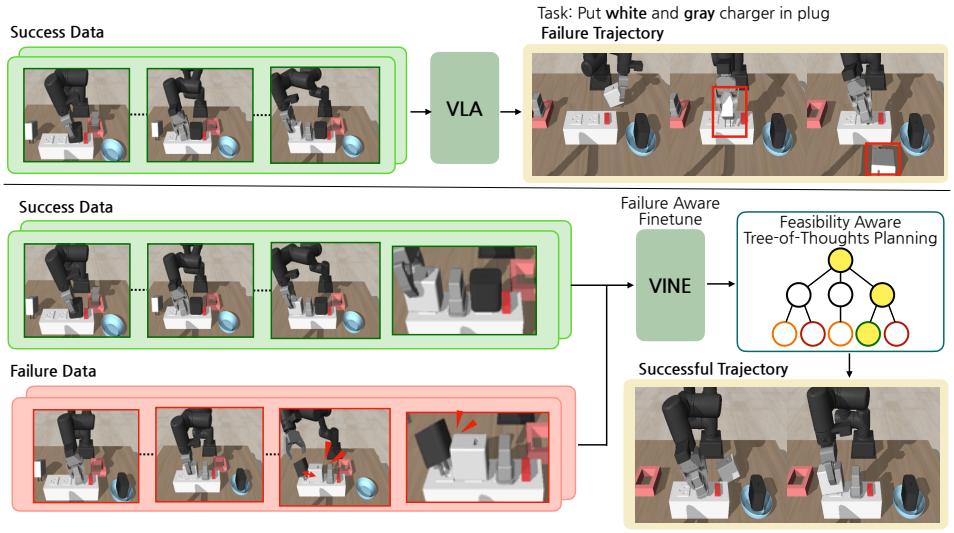

Figure 1: Standard VLAs trained only on success data may produce infeasible trajectories. VINE leverages both success and failure trajectories with a feasibility-aware Tree-of-Thoughts planner, yielding more reliable executions.

Abstract: In this paper, we study how to improve the robustness of Vision–Language–Action (VLA) models by leveraging failure data. Existing VLAs are often trained on successful demonstrations and make limited use of failures, which can yield trajectories that appear plausible yet execute unreliably under variations. We introduce VINE, a dual–system framework in which a Tree-of-Thoughts planner (System2) is finetuned with both success and failure trajectories to estimate reasoning-level feasibility, while a visuomotor controller (System1) executes subgoal actions. Experiments on plug insertion show that incorporating failure-aware value learning improves success rates, especially in unseen settings, surpassing unified VLA baselines and few-shot VLM planners. Our results highlight failure data as an essential yet underutilized resource for enhancing safety and robustness in embodied reasoning.

Keywords: Vision-Language Action Model, Failure Data, Reasoning

1 Introduction

Vision–Language–Action (VLA) models [1, 2, 3, 4, 5] map language and vision to robot actions, offering a unified route from instruction following to control. To enhance the representation of this mapping, recent work integrates chain-of-thought (CoT) prompting [6], exposing intermediate descriptions—scene attributes, spatial relations, and candidate subgoals—before low-level execution [7, 8, 9, 10]. One of the remaining challenges is reliability: we need a robust agent that favors

Workshop on Safe and Robust Robot Learning for Operation in the Real World at CoRL 2025.

trajectories that remain stable under variation and avoids paths prone to failure. This motivates *failure-aware reasoning*: we encourage to use of evidence from both successes and failures to bias selection toward robust plans and away from paths that are prone to fail.

A crucial but often overlooked element in training VLAs is the use of *failure data* [11, 12, 13]. Robot datasets, especially those collected through human teleoperation, naturally include a substantial number of failed attempts, such as unstable grasps, collisions, or incomplete trajectories. These are typically discarded as noise, yet they carry essential information about infeasible transitions and unsafe behaviors. Incorporating failures during training allows models to distinguish between successful and unsuccessful strategies, improving both generalization and robustness. By learning not only from what works but also from what fails, agents can better anticipate risky outcomes and avoid repeating unsafe patterns at deployment.

The open question is how to inject this failure signal into the reasoning loop. We address this by interpreting CoT for embodiment as a *chain of language-described state transitions* (state $\rightarrow$ next-state). In this view, each thought is a hypothesis about how the scene will change if a subgoal is attempted. Failure examples provide counterevidence: they mark transitions that tend to produce collisions, unstable contacts, or unrecoverable poses. Training with both successes and failures calibrates the model's confidence over these hypothesized transitions, discouraging steps that are linguistically plausible yet physically unlikely. Consequently, the chain privileges transitions that are both interpretable in language and empirically likely to succeed.

To this end, we introduce **VINE**, **V**ision-language-action with **IN**tegrated failure-aware r**E**asoning. Our goal is not to design a new backbone; we focus on finetuning existing VLA models to endow them with failure-aware reasoning. We adopt a dual-system design: *System 1* predicts executable action chunks, while *System 2* builds and evaluates a tree of thought states. Unlike prior work that relies only on successful trajectories, System 2 is finetuned with both success and failure data to score candidate subgoals and enforce planning-level feasibility.

Our contributions are twofold. First, we introduce a dual-system framework in which System 2 uses both success and failure data to enforce planning-level safety while System 1 executes grounded actions. Third, we empirically demonstrate that incorporating failure-aware reasoning significantly improves robustness and success rates on challenging manipulation tasks. Together, these results highlight the importance of failure data as a unique and underutilized resource for enhancing the reasoning capabilities of VLAs.

2 Related Work

Vision Language Action Models (VLA) Vision–Language–Action (VLA) models unify perception, language, and action through end-to-end learning [14, 15, 4, 16, 17, 18]. Recent advances include OpenVLA [5, 19] for scalable cross-embodiment training, π_0 and $\pi_{0.5}$[3, 2] with PaliGemma backbones for continuous actions, and GR00T[1], DexVLA [20], and RDT–1B for dual-system, diffusion, and large-scale designs. Yet, these models largely omit explicit reasoning to anticipate failures or verify feasibility, motivating reasoning-augmented VLAs.

Reasoning in VLA Recent work augments VLAs with explicit reasoning to boost generalization and interpretability: ECoT introduces multimodal textual traces and ECoT-Lite preserves most gains with lower supervision [7, 8]. Other modalities include image-based subgoal prediction (CoT-VLA) and structured affordance chains (CoA-VLA) [9, 10], alongside tighter plan–act coupling or latent-plan reinforcement (OneTwoVLA, ThinkAct, MolmoAct) and hierarchical control (Hi-Robot) [21, 22, 23, 24]. Despite these gains, most methods remain success-driven and underuse failure trajectories; we instead exploit *failure-aware reasoning* to calibrate value and improve robustness.

Tree of Thoughts While CoT-based VLAs organize reasoning as a linear sequence of subgoals, *Tree-of-Thoughts (ToT)* [25] casts it as searches over branching thought states. In practice, ToT commonly leverages MCTS [26, 27, 28, 29], with nodes as partial reasoning states and rollouts

estimating success. Recent work emphasizes value guidance—e.g., Liu et al. [30] shows that the PPO-trained value model should guide decoding via MCTS (PPO-MCTS) to avoid prematurely pruning useful candidates. For embodied agents, ToT enables evaluating alternative futures and selecting safer actions, but it hinges on well-calibrated value estimates that success-only data cannot provide. Failure trajectories supply essential negative evidence to separate feasible from infeasible branches, making them critical for value learning.

3 Problem Formulation & Dataset

Objective. Given an imitation dataset $\mathcal{D} = (s_t, a_t, r_t)$ consisting of multimodal states s_t (images and proprioception), low–level actions a_t, and sparse rewards $r_t \in 0, 1$ indicating whether the trajectory reached a success set, our objective is to learn a policy π_θ that maximizes the probability of generating successful trajectories. Unlike conventional imitation learning, which often conditions only on successful rollouts, we leverage both successful and failed trajectories to learn value–aware reasoning that discriminates feasible subgoal sequences from those prone to failure.

For reasoning, we introduce an abstraction function $\phi : s \mapsto n$ that maps concrete states s to high–level *nodes* n (e.g., scene–graph snapshots or key images). Directed *edges* (n_i, n_j) represent hypothesized subgoal transitions between abstract states. A candidate plan is thus a path $z_{0:K} = (n_0, e_0, n_1, \ldots, n_K)$ through this graph, starting from $n_0 = \phi(s_0)$ and ending at a terminal node n_K. We also define the *reasoning–level feasibility* of a path via the success–before–failure probability with success/fail set $\mathcal{G}, \mathcal{F}$

$$\mathsf{Fail}(z_{0:K}) = \Pr\big(\text{terminate in } \mathcal{F} \;\big|\; z_{0:K}, \ell\big), \qquad \mathsf{Succ}(z_{0:K}) = 1 - \mathsf{Fail}(z_{0:K}).$$

With the entrance-reward $r(s, a, s') = \mathbb{I}\{s' \in \mathcal{G}\}$, the undiscounted return equals the success indicator, hence

$$\mathsf{Succ}(z_{0:K}) = \Pr\big(\text{terminate in } \mathcal{G} \;\big|\; \pi\big) = \mathbb{E}_\pi\Big[\sum_{t\geq 0} r_t\Big]$$

i.e., Succ is the value of the policy that follows $z_{0:K}$.

Dataset. Our dataset is composed of human–demonstrated trajectories with both successful and failed outcomes. Each trajectory in $\mathcal{D}$ includes: (i) raw streams $(s_{0:T}, a_{0:T-1}, \ell)$ and terminal r; (ii) a subgoal segmentation $\{[t_k, t_{k+1})\}_{k=0}^{K-1}$ derived by a temporal heuristic based on gripper open and close commands; (iii) Node–edge annotations. Each node is a 2D scene graph extracted at subgoal boundary t_k comprising: (a) the gripper's image–plane position (projection of the measured 3D pose), (b) object bounding boxes proposed by GroundingDINO [31] and named/refined by the multimodal LLM Gemini-2.5-flash [32], and (c) pairwise spatial relations between entities. Directed edges connect consecutive subgoals $k \to k+1$ (e.g., `grasp-x`/`insert-x`).

4 Proposed Method

We implement reasoning-level feasibility with a dual-system agent: **System 2 (Reasoning and Planning)** explores the thought graph, proposes candidate reasoning paths, estimates their success probabilities, and selects the most reasoning-safe plan; **System 1 (Action Modeling)** then executes the next subgoal from the chosen plan with low-level actions

4.1 Architecture

Our agent has two coupled systems. *System 2* operates on a thought graph $\mathcal{T} = (\mathcal{N}, \mathcal{E})$ to propose and score abstract plans (thought paths). *System 1* is a visuomotor controller that executes the selected subgoal chunk from the current node. Each planning cycle (i) builds a tree over $\mathcal{T}$, (ii) selects a leaf by success probability, then (iii) executes one subgoal and updates the current one with the observation. To satisfy real-time requirements, the tree is constructed *before* execution.

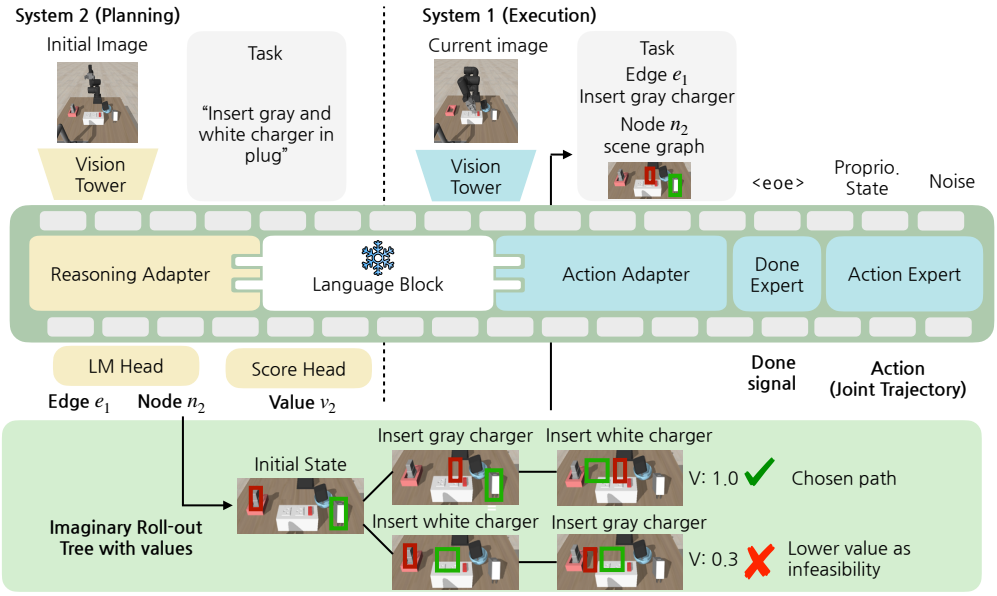

Figure 2: **Overall pipeline.** System 2 plans via tree-of-thoughts, predicting success values to select the highest-value path (green). System 1 executes the chosen subgoal as action chunks with a done signal. Both share a vision–language backbone with adapters.

We build on π_0 [3] as a multimodal backbone that encodes (o, ℓ, a) into a unified token sequence. Because π_0 targets action generation rather than text, we merge its shared layers with a PaliGemma [33] language–vision trunk, yielding strong text generation for thought steps while preserving π_0's action priors. Both systems add LoRA [34] adapters to this shared backbone. **System 1** feeds adapted features to (i) an *action expert* for low-level motor chunks and (ii) a *done expert* that detects subgoal completion. **System 2** uses (i) a language-modeling head to autoregress node/edge text and (ii) a scalar value head to predict leaf success. This shares perception–language representations while keeping reasoning and control decoupled.

4.2 System 2

System 2 instantiates Tree-of-Thoughts (ToT) with three components: (1) discrete thought steps (nodes/edges), (2) candidate generation for nodes and edges, and (3) leaf evaluation by a value function estimating success probability. Reasoning-level safety emerges by selecting the leaf with the highest estimated probability of eventual success, avoiding risks of failure.

Thought steps. Reasoning steps k are distinct from execution time t. Let t_k be the control time when the k-th thought is formed. The node n_k is the grounded scene graph extracted from (ℓ, o_{t_k}), encoding gripper 2d poses, object 2d poses, and relations. The edge e_k is a verifiable subgoal. including $\langle \texttt{done} \rangle$), proposing transition.

Candidate generation. We model next-node and next-edge proposals with a conditional language model (*decision model* P_θ^{dec}) that scores tokens autoregressively.

$$p_\theta(n_{k+1} \mid z_{0:k}, \ell, o_0) = \prod_{j=1}^{N_c} P_\theta^{\text{dec}}(w_j \mid w_{<j}, z_{0:k}, \ell, o_0)\,, \tag{1}$$

$$p_\theta(e_{k+1} \mid n_{k+1}, z_{0:k}, \ell, o_0) = \prod_{j=1}^{N_e} P_\theta^{\text{dec}}(u_j \mid u_{<j}, n_{k+1}, z_{0:k}, \ell, o_0)\,. \tag{2}$$

We train with a standard LM objective over demonstrations $\mathcal{D}$ containing reasoning traces:

$$\mathcal{L}_{\text{LM}} = -\mathbb{E}_{(n_k, e_k, z_{0:k-1}, \ell, o_0)\sim\mathcal{D}} \left[\log p_\theta(n_k \mid z_{0:k-1}, \ell, o_0) + \log p_\theta(e_k \mid n_k, z_{0:k-1}, \ell, o_0)\right]. \tag{3}$$

Node evaluation. We denote the *reasoning state* at thought step k as

$$x_k = (n_k, z_{0:k-1}),$$

the current node together with the reasoning history $z_{0:k-1} = [n_0, e_0, \ldots, n_{k-1}]$. Each chunk e_k terminates in either the success symbol `G`, the failure symbol `F`, or a continuation state $(n_{k+1}, z_{0:k}) \in \mathcal{N} \times \mathcal{Z}$. Thus the node reward is $\bar{R}_k = \mathbf{1}\{x_{k+1} = \texttt{G}\}$, where `G` is a goal state. The value head V_θ is trained to predict the expected thought step reward, i.e. the probability that the next reasoning state ends in success:

$$V_\theta(n_k, z_{0:k-1}) \;\approx\; \mathbb{E}[\bar{R}_k \mid n_k, e_k, z_{0:k-1}] = \Pr(x_{k+1} = \texttt{G} \mid (n_k, z_{0:k-1}), e_k).$$

We train V_θ with first-exit bootstrapping: for each transition $(n_k, e_k, n_{k+1}, r_{k+1})$,

$$y_k = \begin{cases} 1, & \text{if the path succeeds and } e_k = \langle\texttt{done}\rangle, \\ 0, & \text{if the path fails at } k{+}1, \\ \gamma\, V_{\theta'}(n_{k+1}, z_{0:k}), & \text{otherwise,} \end{cases} \tag{4}$$

with $\gamma{=}0.9$ and a target network θ' updated by EMA. The loss is

$$\mathcal{L}_{\text{val}} \;=\; \mathbb{E}\Big[\big(V_\theta(n_k, z_{0:k-1}) - y_k\big)^2\Big].$$

In inference, V_θ is evaluated only at leaf nodes, and the planner selects the leaf with maximal value, i.e., the thought path most likely to succeed.

Search Algorithm We leverage the MCTS [35] style algorithm to generate the tree. Track visit counts N, cumulative returns W, and means $Q = W/\max(1, N)$. The decision model proposes K edges per node; we min–max normalize to priors $\tilde{P}(e \mid n, z)$. Rank edges by a convex blend

$$s(n, e) = \alpha\, Q(n) + (1 - \alpha)\, \tilde{P}(e \mid n, z_{<n}).$$

Per iteration: pick the top-M leaves by Q; from each, expand the top-B edges by s, create successors n', *evaluate leaves only* with $v = V_\theta(n', z_{0:t}) \in [0, 1]$, and backpropagate $N{\leftarrow}N{+}1$, $W{\leftarrow}W{+}v$, $Q{\leftarrow}W/N$. Terminate on budget and return the plan to

$$\hat{n} = \arg\max_{n \in \mathcal{L}_e} V_\theta(n, z_{\prec n}), \qquad \hat{z} = \text{path}(n_0 \rightsquigarrow \hat{n}).$$

Since we evaluate leaves only, maximizing V_θ at the leaf directly optimizes the estimated success probability.

4.3 System 1

Given observations o_t, instruction ℓ, current edge e_k and successor node n_{k+1}, System 1 outputs an action chunk $\mathbf{A}_t \in \mathbb{R}^{H \times d_a}$ and a termination probability for `<eoe>`, end-of-edge. We train System 1 on success data. Both the action and done experts read from the shared backbone, but no cross-attention or feature exchange between the two experts.

System 1 uses an *action expert* $\mathbf{v}_\theta$ trained via conditional flow matching: with Gaussian path $\mathbf{A}_t^\tau = \alpha(\tau)\mathbf{A}_t + \sigma(\tau)\boldsymbol{\epsilon}$,

$$L_\tau = \mathbb{E}_{\tau, \boldsymbol{\epsilon}} \big\| \mathbf{v}_\theta(\mathbf{A}_t^\tau, o_t, \ell, e_k, n_{k+1}) - \mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t) \big\|_2^2.$$

A *done expert* predicts subgoal termination $p_\theta^{\text{done}}(\texttt{<eoe>} \mid e_k, n_{k+1}, o_t, \ell)$ with focal loss

$$\mathcal{L}_{\text{focal}} = -\alpha_d\, (1 - p_\theta^{\text{done}})^{\gamma_d} \log p_\theta^{\text{done}}.$$

The joint objective is

$$\mathcal{L}_{\text{S1}} = \lambda_{\text{flow}} \mathbb{E}_\tau[L_\tau] + \lambda_{\text{eoe}} \mathcal{L}_{\text{focal}}.$$

At inference, initialize $\mathbf{A}_t^0 \sim \mathcal{N}(0, \sigma_0^2 I)$ and integrate $\mathbf{A}_t^{\tau+\delta} = \mathbf{A}_t^\tau + \delta\, \mathbf{v}_\theta(\mathbf{A}_t^\tau, o_t, \ell, e_k, n_{k+1})$ for $\tau \in \{0, \delta, \ldots, 1 - \delta\}$ to obtain $\widehat{\mathbf{A}}_t$; the done head then emits `<eoe>` or continues the current edge.

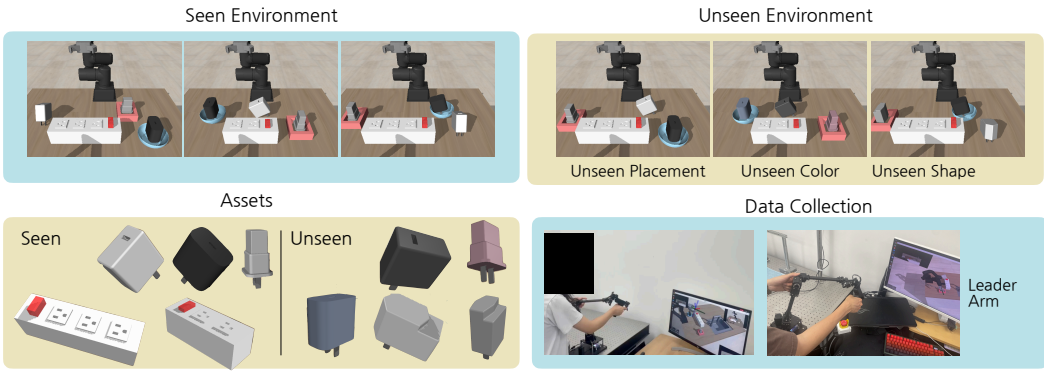

Figure 3: Plug insertion environments and data collection. We evaluate in both seen and unseen settings (placement, color, shape) with corresponding assets, and collect demonstrations via a leader–follower teleoperation setup.

5 Experiments

We evaluate VINE on the manipulation task, plug insertion under seen/unseen splits, reporting Success Rate (SR). Our study tests the following hypotheses:

H1 (Reasoning-level feasibility). A learned, failure-aware value in System 2, coupled with tree search, improves robustness on *unseen* settings versus unified VLAs and prompted VLM planners.

H2 (Role of failure data). Training System 2 with both success and failure trajectories yields higher SR than (i) success-only training and (ii) few-shot failure prompting in VLMs, indicating that explicit value learning provides stronger generalization than textual patterning.

5.1 Environment Setup & Baselines

Environment. The environment is built upon a MuJoCo simulator. Plug insertion (seen: 9 configs) combines three table placements with 2- or 3-socket strips and orientation variants; *unseen* splits introduce novel charger placement, unseen strip color, and unseen charger shape. The strip collision margin is slightly relaxed to reduce teleop burden. Control runs at 20 Hz with joint-position actions.

Data collection. We gather human teleoperation trajectories via a leader–follower setup (Fig. 3). Plug insertion consists of 450 demonstrations collected over predefined insertion orders (six paths for 3-socket and two for 2-socket), balanced per path. Each trajectory is labeled as success or failure, with outcomes naturally stochastic: for instance, PlugStrip-3 Fail dominates (38.9%, 175/450), followed by PlugStrip-2 Success (30.4%, 137), PlugStrip-3 Success (21.1%, 95), and PlugStrip-2 Fail (9.6%, 43). This yields a multi-modal, diverse dataset where success rates vary across insertion orders, reflecting realistic execution variability.

Baselines To assess our two-system framework, we compare against three groups: (i) unified VLA models, (ii) VLM-as-planner baselines that replace System2 while keeping System1 fixed, and (iii) our System 2 variants (Chain/Tree/Full) to isolate branching and failure-conditioned value. Since our environment operates at 20Hz in a joint-position action representation, we select baselines that can be executed under this control frequency and action format.

- **Unified VLA models.** OpenVLA-OFT [19], GR00T N1.5 [1], and π_0 [3], plus a reward-conditioned π_0 variant using failure data [36].
- **VLM-as-System 2.** Replace our planner with SOTA VLMs (e.g., GPT-4o [37], Gemini-2.5-Flash [32]) using few-shot prompting to propose subgoals/next scene graphs, with and without failure examples; System 1 (our action executor) is fixed.

Table 1: Results. Success Rates on Plug Insertion.

Models	Failure Data	Plug Insertion Seen	Unseen	Average
Unified Model				
OpenVLA-OFT [19]	×	0.244	0.044	0.144
GR00T N1.5 [1]	×	0.422	0.244	0.333
π_0 [3]	×	0.689	0.267	0.477
π_0 [3] + Reward Cond. [36]	✓	0.489	0.111	0.300
SOTA VLM as System2				
GPT-4o [37]	×	0.733	0.311	<u>0.522</u>
GPT-4o [37]	✓	0.711	<u>0.333</u>	0.488
Gemini-2.5-Flash [32]	×	<u>0.756</u>	0.200	<u>0.522</u>
Gemini-2.5-Flash [32]	✓	0.711	0.289	0.500
Role of failure data in Our System2				
VINE-Chain	×	0.733	0.244	0.488
VINE-Tree	×	0.711	0.289	0.500
VINE-Full (Ours)	✓	**0.800**	**0.422**	**0.611**

- **Our System 2 variants.** VINE-Chain (no branching without failure data), VINE-Tree (tree search without failure data, scoring with confidence), and VINE-Full (tree search + failure-conditioned value).

5.2 Results

H1. Reasoning-level feasibility vs. baselines. VINE -Full delivers the strongest unseen robustness and the highest averages on plug (Table 1). On plug–unseen, VINE outperforms the best VLM planner with failure prompts, GPT-4o [37] by $+0.089$ ($+26.7\%$ rel.), and π_0 by $+0.155$ ($+58.1\%$ rel.). On plug–avg, VINE attains 0.611 vs. the best non-ours 0.522 ($+0.089$, $+17.1\%$ rel.). Notably, unified VLAs overfit to seen settings (e.g., π_0: $0.689 \rightarrow 0.267$ on plug). Reward-conditioning π_0 hurts plug performance (seen -0.20, unseen -0.156), suggesting that injecting failure via scalar rewards at the state level does not impart the global, plan-level risk signals needed for generalization.

H2. Value from failure data. Failure-conditioned training of the value head is pivotal. VINE improves plug–unseen from 0.244 (Chain) / 0.289 (Tree) to 0.422 and plug–avg from $0.488/0.500$ to 0.611. Negative demonstrations calibrate the value and prune unsafe paths. By contrast, confidence-only signals (LM likelihoods or success-only value) are often miscalibrated and overfit to seen/language regularities, yielding overconfident yet unsafe plans (plug–unseen stays at $0.244/0.289$ for Chain/Tree). Few-shot failure prompting helps (plug $0.311 \rightarrow 0.333$, Gemini-2.5-flash [32] $0.200 \rightarrow 0.289$) but still trails learned failure-aware value. We attribute this to VLM planners relying on language-only cues rather than dynamics-grounded supervision, producing linguistically plausible yet physically infeasible subgoals. Hence, a learned, failure-aware value is necessary to reliably prune unsafe branches and achieve higher robustness under shift.

5.3 Ablation Studies

We ablate backbone weight merging between PaliGemma [33] and the π_0 [3] trunk. Specifically, we linearly interpolate the shared layers,

$$\theta_g^{\text{merge}}(\lambda_m) = \lambda_m \, \theta_g^{\pi_0} + (1 - \lambda_m) \, \theta_g^{\text{PG}},$$

while keeping non-overlapping heads separate and training LoRA on top.

Table 2: Ablation on weight merging between PaliGemma and π_0.

λ	Seen SR.	Unseen SR.	Avg. SR.
0.5	0.756	0.356	0.555
0.6	**0.800**	**0.422**	**0.611**
0.7	0.733	0.289	0.511
0.8	0.422	0.244	0.333

Table 2 shows the effect of varying interpolation weight λ. Performance peaks near λ_m=0.6 (Avg. 0.611, Unseen 0.422). Larger λ_m (0.7–0.8) overweights π_0, hurting generalization, while smaller λ_m (0.5) overweights PaliGemma, weaken-

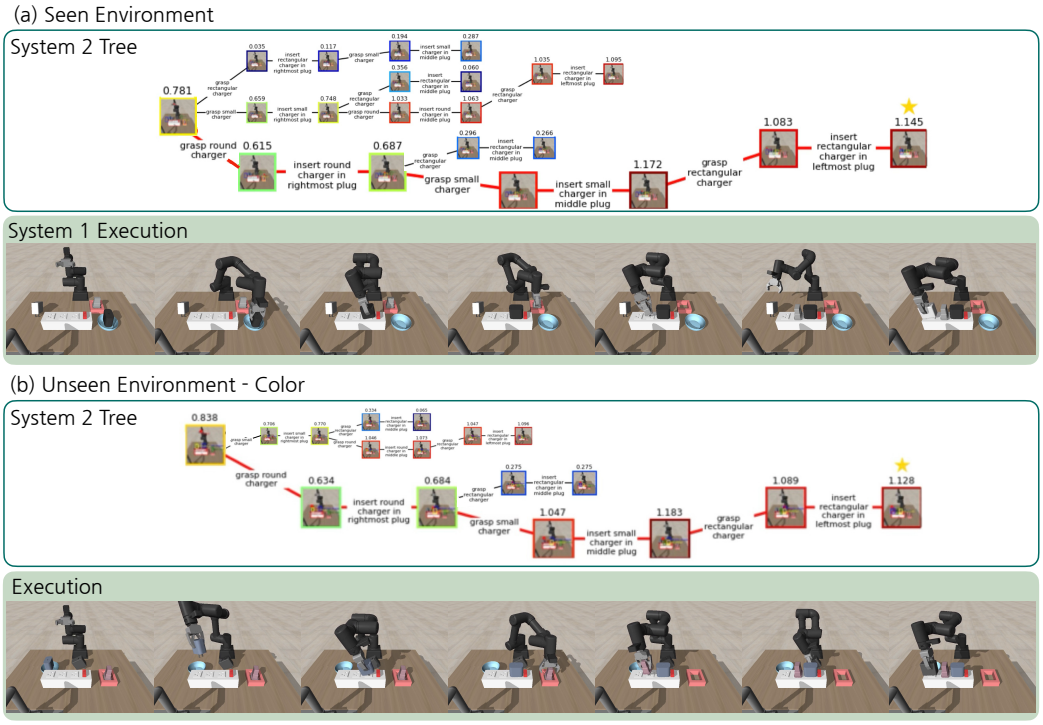

Figure 4: Qualitative results in seen and unseen environments. System2 builds a failure-aware planning tree with feasibility scores, and System1 executes the selected subgoals, producing successful trajectories.

ing low-level action modeling. This balance highlights the need to align linguistic plausibility with feasibility. The qualitative results with trajectories and trees are demonstrated in Figure 4.

5.4 Limitations

Subgoal-level planning without mid-chunk correction or anytime updates allows disturbances between replans; failure-prompted VLM planners narrow our margin, and because System 1 is trained only on successes it struggles to recover from low-level mistakes. In addition, representing states as an abstract 2d scene graph may not be enough to fully capture the dynamics, leading to the requirements of more precise world models. Anytime replanning, failure-aware action learning (e.g., offline RL), and distillation/caching to cut runtime would improve calibration and speed.

6 Conclusion

We introduced VINE, a dual–system VLA for *reasoning-level feasibility* that pairs a fast visuomotor controller (System 1) with a failure-aware Tree-of-Thoughts planner (System 2). System 2 learns a calibrated success value from successful and failed trajectories and uses MCTS to select plans; System 1 executes the chosen subgoal via action chunks. VINE consistently improves robustness over unified VLA baselines and matches or exceeds prompted SOTA VLM planners, with the largest gains in unseen environments. By modeling failure at the level of plans rather than individual actions, VINE aims to improve robustness and auditability while reducing unsafe executions and hardware wear. Remaining risks include value miscalibration under shift, motivating calibration checks, human oversight, and runtime safety monitors in safety-critical settings.

References

[1] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. 1, 2, 6, 7

[2] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054. 1, 2

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. 1, 2, 4, 6, 7

[4] O. X.-E. Collaboration, A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Kolobov, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Büchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. V. Frujeri, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Yang, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. B. Amor, H. I. Christensen, H. Furuta, H. Bharadhwaj, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Vakil, J. Bohg, J. Bingham, J. Wu, J. Gao, J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silvério, J. Hejna, J. Booher, J. Tompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Lekkala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, M. Z. Irshad, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Mart'in-Mart'in, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Tulsiani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Kumar, V. Vanhoucke, V. Guizilini, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Pang, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Dou, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, Z. Fu, and Z. Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023. 1, 2

[5] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. 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. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024. URL https://openreview.net/forum?id=ZMnD6QZAE6. 1, 2

[6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1

[7] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. In *Proc. of the 8th Annual Conference on Robot Learning (CoRL)*, 2024. URL https://openreview.net/forum?id=S70MgnIA0v. 1, 2

[8] W. Chen, S. Belkhale, S. Mirchandani, O. Mees, D. Driess, K. Pertsch, and S. Levine. Training strategies for efficient embodied reasoning. *arXiv preprint arXiv:2505.08243*, 2025. 1, 2

[9] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proc. of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 1702–1713, 2025. 1, 2

[10] J. Li, Y. Zhu, Z. Tang, J. Wen, M. Zhu, X. Liu, C. Li, R. Cheng, Y. Peng, and F. Feng. Improving vision-language-action models via chain-of-affordance. *arXiv preprint arXiv:2412.20451*, 2024. 1, 2

[11] Z. Liu, A. Bahety, and S. Song. REFLECT: Summarizing robot experiences for failure explanation and correction. In *Proc. of the 7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=8yTS_nAILxt. 2

[12] M. Diehl and K. Ramirez-Amaro. Why did i fail? a causal-based method to find explanations for robot failures. *IEEE Robotics and Automation Letters*, 7(4):8925–8932, 2022. 2

[13] S. Choi, K. Lee, and S. Oh. Robust learning from demonstrations with mixed qualities using leveraged gaussian processes. *IEEE Transactions on Robotics*, 35(3):564–576, 2019. doi: 10.1109/TRO.2019.2891173. 2

[14] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. In *Proc. of the Robotics: Science and System (RSS)*, 2023. 2

[15] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, B. Ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proc. of The 7th Conference on Robot Learning (CoRL)*, pages 2165–2183, 2023. 2

[16] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy. In *Proc. of the Robotics: Science and Systems (RSS)*, Delft, Netherlands, 2024. 2

[17] J. Wen, Y. Zhu, M. Zhu, Z. Tang, J. Li, Z. Zhou, X. Liu, C. Shen, Y. Peng, and F. Feng. Diffusionvla: Scaling robot foundation models via unified diffusion and autoregression. In *Proc. of the Forty-second International Conference on Machine Learning (ICML)*, 2025. 2

[18] T. L. Team, J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina, N. Kuppuswamy, K.-H. Lee, K. Liu, D. McConachie, I. McMahon, H. Nishimura, C. Phillips-Grafflin, C. Richter, P. Shah, K. Srinivasan, B. Wulfe, C. Xu, M. Zhang, A. Alspach, M. Angeles, K. Arora, V. C. Guizilini, A. Castro, D. Chen, T.-S. Chu, S. Creasey, S. Curtis, R. Denitto, E. Dixon, E. Dusel, M. Ferreira, A. Goncalves, G. Gould, D. Guoy, S. Gupta, X. Han, K. Hatch, B. Hathaway, A. Henry, H. Hochsztein, P. Horgan, S. Iwase, D. Jackson, S. Karamcheti, S. Keh, J. Masterjohn, J. Mercat, P. Miller, P. Mitiguy, T. Nguyen, J. Nimmer, Y. Noguchi, R. Ong, A. Onol, O. Pfannenstiehl, R. Poyner, L. P. M. Rocha, G. Richardson, C. Rodriguez, D. Seale, M. Sherman, M. Smith-Jones, D. Tago, P. Tokmakov, M. Tran, B. V. Hoorick, I. Vasiljevic, S. Zakharov, M. Zolotas, R. Ambrus, K. Fetzer-Borelli, B. Burchfiel, H. Kress-Gazit, S. Feng, S. Ford, and R. Tedrake. A careful examination of large behavior models for multitask dexterous manipulation. 2025. URL https://arxiv.org/abs/2507.05331. 2

[19] M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. 2, 6, 7

[20] J. Wen, Y. Zhu, J. Li, Z. Tang, C. Shen, and F. Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025. 2

[21] F. Lin, R. Nai, Y. Hu, J. You, J. Zhao, and Y. Gao. Onetwovla: A unified vision-language-action model with adaptive reasoning. *arXiv preprint arXiv:2505.11917*, 2025. 2

[22] C.-P. Huang, Y.-H. Wu, M.-H. Chen, Y.-C. F. Wang, and F.-E. Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025. 2

[23] J. Lee, J. Duan, H. Fang, Y. Deng, S. Liu, B. Li, B. Fang, J. Zhang, Y. R. Wang, S. Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025. 2

[24] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025. 2

[25] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Proc. of the Advances in neural information processing systems (NeurIPS)*, 36:11809–11822, 2023. 2

[26] Z. Zhao, W. S. Lee, and D. Hsu. Large language models as commonsense knowledge for large-scale task planning. In *Proc. of the Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL https://openreview.net/forum?id=Wjp1AYB8lH. 2

[27] D. Zhang, S. Zhoubian, Z. Hu, Y. Yue, Y. Dong, and J. Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *Proc. of the Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=8rcFOqEud5. 2

[28] Z. Gao, B. Niu, X. He, H. Xu, H. Liu, A. Liu, X. Hu, and L. Wen. Interpretable contrastive monte carlo tree search reasoning, 2025. URL https://openreview.net/forum?id=F4f1afsm3R. 2

[29] Y. Chi, K. Yang, and D. Klein. ThoughtSculpt: Reasoning with intermediate revision and search. In L. Chiruzzo, A. Ritter, and L. Wang, editors, *Proc. of the Findings of the Association for Computational Linguistics (NAACL)*, pages 7685–7711, Albuquerque, New Mexico, Apr. 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi:10.18653/v1/2025.findings-naacl.428. URL https://aclanthology.org/2025.findings-naacl.428/. 2

[30] J. Liu, A. Cohen, R. Pasunuru, Y. Choi, H. Hajishirzi, and A. Celikyilmaz. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding. In *Proc. of the First Conference on Language Modeling (CoLM)*, 2024. URL https://openreview.net/forum?id=kh9Zt2Ldmn. 3

[31] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *Proc. of the European conference on computer vision (ECCV)*, pages 38–55. Springer, 2024. 3

[32] G. Comanici, E. Bieber, M. Schaekermann, I. Pasupat, N. Sachdeva, I. Dhillon, M. Blistein, O. Ram, D. Zhang, E. Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 3, 6, 7

[33] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024. 4, 7

[34] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9. 4

[35] D. Silver, A. Huang, C. J. Maddison, and et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 5

[36] H. Yuan, K. Huang, C. Ni, M. Chen, and M. Wang. Reward-directed conditional diffusion: Provable distribution estimation and reward improvement. In *Proc. of the Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL https://openreview.net/forum?id=58HwnnEdtF. 6, 7

[37] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 6, 7