ONE PHASE IS ENOUGH: Sim-to-Real Transfer with Direct Visual Reinforcement Learning

John Marangola*[1,2] Antonia Bronars[1,2] Younghyo Park[1,2]
Stephen Andrews[1,2] Kushal Kedia[1,2] Pulkit Agrawal[1,2]

[1]Massachusetts Institute of Technology [2]Improbable AI Lab

***Abstract*— Sim-to-real reinforcement learning (RL) has enabled impressive robotic behaviors, yet deploying policies directly from vision remains challenging. A common approach first trains a teacher policy with RL using privileged state observations, then distills its behavior into a vision-based student via imitation. However, the resulting student can substantially underperform its teacher. Directly training visual policies with RL offers a promising alternative, but learning task-relevant visual representations from scratch can be sample-inefficient, making one-phase visual RL intractable.**

We propose DRIFT, a simple recipe for efficient one-phase visual RL using image observations available at test time. Our key insight is to use frozen, pretrained foundation vision encoders, allowing RL to focus on learning control from existing visual representations. Freezing the encoder avoids costly backpropagation through the vision backbone, while its pretrained representations enable real-world transfer from fast, rasterized simulation renders. Under matched architectures, observations, and compute budgets, DRIFT outperforms two-phase distillation and achieves a 58.8% relative improvement in average real-world success rate across three manipulation tasks. Controlled simulation evaluations demonstrate that DRIFT's relative advantage grows from roughly 9% in-distribution to over 61% under out-of-distribution shifts, suggesting that stronger generalization contributes to superior real-world transfer.

I. INTRODUCTION

Sim-to-real reinforcement learning (RL) provides a scalable alternative to real-world data collection by enabling massive amounts of parallel trial-and-error in simulation. This paradigm has pushed robot hardware toward its performance limits in locomotion [1]–[4] and dexterous manipulation [5]–[7]. However, these successes largely rely on compact state observations, such as object poses, which are estimated through heavily engineered perception pipelines.

To deploy policies directly from visual observations, a common approach is to distill privileged state-based teacher policies into image-based students [8]. The teacher is first trained with RL using privileged simulator state, after which its behavior is transferred to the visual student via imitation. However, this two-phase process introduces two sources of performance degradation. First, the privileged teacher may rely on information unavailable to the student, creating a realizability gap that prevents faithful policy distillation [9]. Second, although imitation-trained students can achieve high success rates in simulation, they may generalize poorly across the sim-to-real gap [10]. Consequently, distilled visual policies can substantially underperform their state-based RL teachers upon real-world deployment [11].

*Corresponding author. Email: jmgola@mit.edu



Fig. 1. **One-phase visual RL closes the realizability gap, generalizes better outside of the training distribution and exhibits superior transfers to the real-world over two-phase approaches.** Policy learning paradigm schematics above; Push-T results below: (a) full-pipeline L40S-equivalent GPU hours, with a 2-hour EMA half-life per seed; (b) Out of distribution evaluation (OOD) in simulation; (c) real-world transfer. Curves show three-seed means ± one standard deviation; OOD bars show three-seed means; real-world bars show separately evaluated policies (30 trials each). Dotted line: privileged teacher performance.

One-phase visual RL offers the appealing alternative of directly optimizing the policy with image observations available at test time. However, this couples the two difficult problems of learning task-relevant visual representations and control policies, with training the vision encoder from scratch typically dominating the computational cost [12]. We address this challenge by building on pretrained foundation vision encoders [13], allowing RL to focus on learning control from existing visual representations. Freezing the encoder eliminates backpropagation through the vision backbone, while its pretrained representations enable policies trained on fast, rasterized simulation renders to transfer to real-world images. Together, these design choices enable one-phase visual RL to outperform two-phase distillation under matched architectures, observations, and compute budgets.

Beyond simulation performance, we investigate how one-phase visual RL and two-phase distillation differ in real-

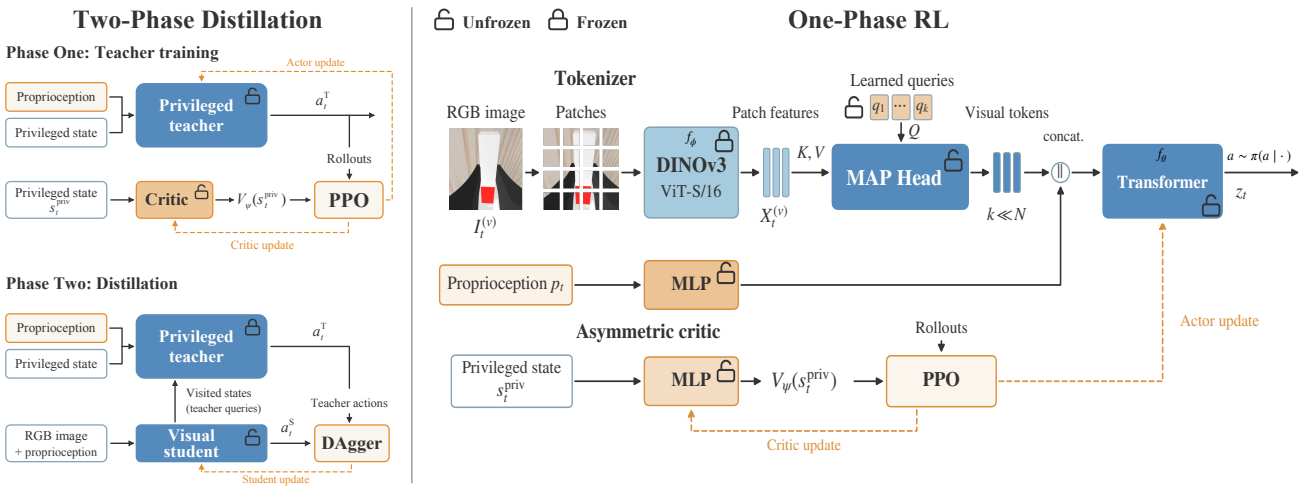

Fig. 2. **DRIFT learns visual policies directly from task rewards.** Simulation provides privileged state information, such as object poses and velocities, that is not directly available at deployment. Two-phase approaches use this information to train a teacher, then train a visual student to imitate its actions (left). We instead optimize the visual policy directly with RL, using a frozen pretrained vision encoder to make training efficient (right).

world transfer and generalization capabilities. ***Our key finding is that even when the two approaches achieve comparable success in simulation, one-phase visual RL transfers substantially better to the real world, achieving a 58.8% relative improvement in average success rate over two-phase distillation across three manipulation tasks.*** To investigate this gap, we evaluate both approaches under controlled in-distribution (ID) and out-of-distribution (OOD) conditions in simulation, varying dynamics, visual appearance, and initial states. One-phase visual RL achieves roughly 9% higher average ID success, consistent with avoiding the realizability gap introduced by distillation from privileged state-based teachers. This relative advantage grows to over 61% under OOD conditions, suggesting that the stronger real-world transfer reflects a broader advantage in generalization beyond the training distribution. Our contributions are:

- We introduce **DRIFT**: ***D**irect **R**einforcement Learning from **I**mages **F**or **T**ransfer*, a simple recipe for efficient one-phase visual RL using frozen foundation vision encoders.
- We demonstrate a generalization gap between one-phase visual RL and two-phase distillation through controlled simulation evaluations, showing that **DRIFT** achieves a 61% relative improvement in OOD settings.
- We validate **DRIFT** across three real-world manipulation tasks, achieving a 58.8% relative improvement in average success rate over two-phase distillation and show generalization to novel object instances unseen during training.

II. Related Work

Teacher-Student Distillation. Privileged-policy distillation has enabled sim-to-real transfer for dexterous grasping, multi-fingered manipulation, and humanoid loco-manipulation [14]–[16]. However, strong teacher performance does not guarantee equally capable students [9]. A typical limitation is information asymmetry (realizability gap), in which teacher actions depend on state that the student cannot infer from its observations [17]. To address this issue, Messikommer et al. [18] jointly train teacher and student policies, adding a teacher–student action-disagreement penalty to the teacher's reward and a supervised loss that aligns the teacher with the student, encouraging behaviors that are realizable from the student's limited observations. Singh et al. [19] train depth-based teachers with RL before distilling them into RGB policies, improving performance over distillation from state-based teachers. Our approach, instead, directly trains the deployed RGB policy with RL. Furthermore, we examine the robustness limitations of distillation beyond the well-established realizability gap.

Efficient Visual RL and Frozen Representations. Direct visual RL has become increasingly tractable with advances in GPU-parallelized simulation and rendering [20], [21]. Prior work improves training efficiency through low-resolution images and lightweight encoders [22], encoder freezing and feature caching [12], and demonstrations that initialize policies and guide RL updates [23]. Frozen pretrained encoders provide another way to simplify visual RL by removing the need to learn visual features from task rewards [24]. Most closely related, PoliFormer [25] combines frozen DINOv2 features with on-policy RL for zero-shot sim-to-real navigation. We build on this approach for contact-rich manipulation, where policies must extract precise spatial information from visual observations. Our recipe combines learned attention pooling over frozen patch features with fast rasterized rendering, reducing the computational burden of visual control while preserving access to spatially detailed representations.

Generalization in RL vs. Supervised Learning. Studies of foundation model post-training find that RL can improve OOD generalization and preserve prior capabilities better than supervised fine-tuning [10], [26]. In robotics, Liu et al. [27] similarly find that RL fine-tuning improves semantic generalization and execution robustness over supervised fine-tuning of VLAs. Whereas these studies examine adaptation of pretrained models, we compare direct visual RL with on-policy teacher distillation under matched architectures,

observations, and compute budgets.

III. Preliminaries

At time t, the robot receives an observation $o_t = (I_t, p_t)$ comprising RGB images I_t and proprioception p_t, and selects a continuous action a_t. A reward r_t specifies the desired task behavior. We compare two ways of learning a visual policy from simulation: directly optimizing its task return with RL, and training it to imitate a privileged RL teacher.

A. Direct Visual Reinforcement Learning

Direct visual RL (Fig. 2, right) learns a policy $\pi_\theta(a_t \mid o_t)$ by maximizing its expected discounted return,

$$J(\theta) = \mathbb{E}_{\tau \sim p_{\pi_\theta}} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right], \qquad \max_\theta J(\theta), \tag{1}$$

where p_{π_θ} is the trajectory distribution induced by the policy in simulation, T is the episode horizon, and $\gamma \in [0, 1]$ is the discount factor. The learning objective therefore evaluates actions through their consequences for task return.

B. Two-Phase Policy Distillation

Two-phase training first learns a teacher $\pi_T(a_t \mid s_t^{\text{priv}})$ using the same expected-return objective, but with privileged simulator state such as object poses and velocities as observations. The second phase learns a visual student $\pi_{S,\theta}(a_t \mid o_t)$ by mimicking the teacher's actions. In the DAgger-style [28] distillation considered here (Fig. 2, left), the current student generates trajectories and the frozen teacher labels the visited states. The student policy is updated via:

$$\theta^{i+1} = \arg\max_\theta \; \mathbb{E}_{\tau \sim p_{\pi_{S,\theta^i}}} \left[\frac{1}{T} \sum_{t=0}^{T-1} \log \pi_{S,\theta}(a_t^T \mid o_t) \right] \tag{2}$$

where $a_t^T = \mu_T(s_t^{\text{priv}})$ is the teacher's mean action, and θ^i denotes the student parameters used to collect trajectories. Although data collection is on-policy, the student is optimized to reproduce teacher actions rather than maximize its own task return.

IV. Method

We propose **DRIFT**, a tractable recipe for one-phase visual RL and zero-shot sim-to-real transfer (Fig. 2). We call the approach "one phase" because it does not rely on a privileged teacher policy, nor expert demonstrations during training. **DRIFT** consists of a frozen ViT encoder, learned attention pooling, an encoder-only transformer actor; it is trained entirely in simulation with asymmetric actor-critic RL [29] (PPO) before being deployed directly in the real world.

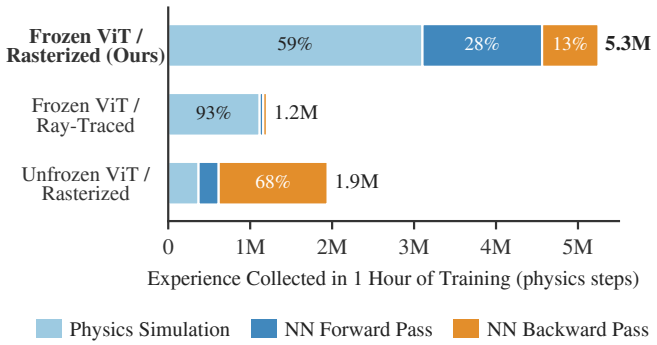

Fig. 3. **Frozen ViT features and rasterization substantially increase training throughput.** Visual RL breakdown per training hour on `Peg-Insertion`. Bar length gives physics steps collected per hour; segments show the fraction of timed training spent in simulation (including rendering), the neural-network forward pass, and backpropagation. Fractions are re-normalized over these three components; evaluation and un-timed overhead are under 2%. Rows compare rasterization with ray tracing and a frozen with an unfrozen DINOv3 ViT-S/16 encoder. Ray tracing makes simulation dominate; unfreezing the encoder makes backpropagation dominate.

A. Algorithmic Decisions for RL Training

We directly train visual policies with distributed Proximal Policy Optimization [30] (PPO) using an asymmetric critic $v_\psi(s_t^{\text{priv}})$ that accesses privileged simulator state during training (Fig. 2, right). The actor is a three-layer encoder-only transformer with hidden dimension 64 that uses bidirectional self-attention. Similar to Dosovitskiy et al. [31], we add learnable position embeddings to tokens by type. Self-attention jointly integrates visual and proprioceptive tokens through a common representation that can accommodate different visual encoders, camera configurations, and per-view token budgets. We parallelize rollouts and synchronize PPO updates across GPU workers, using Adam [32] with a constant base learning rate of 3×10^{-4}.

B. Frozen Visual Features

A frozen encoder f_ϕ produces patch embeddings

$$X_t = \text{sg}[f_\phi(g(I_t))] \in \mathbb{R}^{N \times d_e}, \qquad \nabla_\phi \mathcal{L}_{\text{RL}} = \mathbf{0}, \tag{3}$$

where sg denotes the stop-gradient operator, and g denote the desired crop, resize, and encoder normalization. Using frozen patch embeddings allows us to avoid backpropagating through an expressive visual encoder, which is a primary bottleneck for visual RL training. Freezing the visual encoder also amortizes the forward-pass cost across the K PPO epochs within each policy improvement step. Computing patch embeddings once and reusing them across epochs reduces encoder forward-pass compute by a factor of K, leading to a further increase in training throughput. Together, these advantages yield a substantial increase in the amount of experience **DRIFT** collects in an hour of training (Fig. 3, top), compared with unfreezing the encoder (Fig. 3, bottom).

Furthermore, freezing preserves the visual representations learned through large-scale pretraining on natural images, avoiding encoder updates that could specialize these features to the appearance of simulated images. We find that these

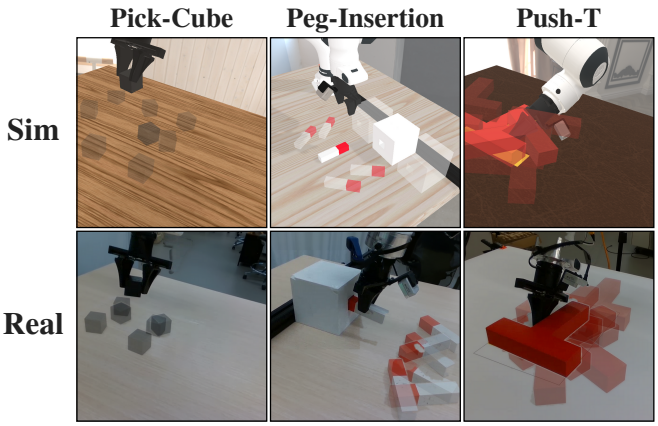

Fig. 4. **Tasks in simulation and the real world.** Initial-state views for Pick-Cube, Peg-Insertion, and Push-T in simulation (top) and the real world (bottom). Translucent objects in the simulation row illustrate different initial poses for the objects. The goal state in the realworld is shown as well.

representations support zero-shot sim-to-real transfer even when policies are trained with fast rasterized images. This allows us to avoid the substantial cost of high-fidelity ray-traced rendering, which also limits the amount of experience that can be collected in an hour of training (Fig. 3, middle).

We find that frozen `[CLS]` features alone are insufficient for precision manipulation, consistent with recent findings [33]. Processing all DINOv3 patch embeddings is computationally expensive. Therefore, we employ a lightweight multihead attention pooling (MAP) head [34]–[36], whose learnable queries cross-attend to the patch embeddings to produce a compact sequence of visual tokens for the transformer policy. This significantly reduces the cost of downstream self-attention which scales quadratically with the number of input tokens.

V. Experimental Setup

A. Task Design

We select three tasks (Fig. 4) spanning simple grasping, non-prehensile pushing, and precision insertion: 1) `Pick-Cube`, grasping and lifting a cube; 2) `Push-T`, aligning a T-shaped block with a visible goal through non-prehensile pushing; and 3) `Peg-Insertion`, inserting a square peg into a hole with 2mm clearance through precise, contact-rich interaction. Object and goal initial poses are randomized across tasks.

B. Training Details

We use ManiSkill [37] with a shared vectorized simulation environment across tasks. The action space is $a = [\delta q, g] \in \mathbb{R}^8$, specifying relative arm joint-position targets and the desired gripper position. Policies observe RGB images and proprioception, $o = (I, p)$, where $p \in \mathbb{R}^{15}$ contains joint positions, joint velocities, and measured gripper position; `Push-T` omits gripper position, yielding $p \in \mathbb{R}^{14}$. Simulated cameras and robot actuator dynamics are calibrated against the real system nominal parameters, as detailed in sections VII-A and VII-B in the appendix. To promote sim-to-real transfer, we randomize object friction and appearance, table textures, lighting through sampled HDR environment maps, and camera extrinsics around the calibrated values. We additionally replace image background pixels using simulator segmentation masks with approximately 27k natural images.

C. Evaluation Protocol

We deploy policies at 20Hz on a Franka Research 3 (FR3) arm, using RGB images from a wrist camera and proprioception observations. We report *success rate* and *throughput*, defined as successful task completions per unit of rollout time, counting successful episodes until first success and failed episodes for the full horizon. For each method and task, we periodically evaluate training checkpoints over 1,024 simulation episodes and retain the best checkpoint from each of three seeds. We select among these checkpoints using five real-world rollouts each, then report the selected checkpoint's performance over 20 or 30 rollouts.

VI. Experimental Results

We answer the following questions:

A. How does **DRIFT** compare with two-phase distillation under in-distribution (ID) and out-of-distribution (OOD) settings in simulation, given matched training compute budgets?
B. Does **DRIFT** transfer more effectively to the real world than two-phase distillation?
C. Which visual encoder yields the best sim-to-real success?

A. In-Distribution and Out-of-Distribution Sim Evaluations

Setup. We compare **DRIFT** against two-phase distillation under matched architectures and compute budgets. Both methods use the same frozen DINO visual encoder and are trained for 150 NVIDIA L40S equivalent GPU hours. We evaluate their robustness to distribution shift on the `Peg-Insertion` and `Push-T` tasks by measuring performance under both in-distribution (ID) and out-of-distribution (OOD) conditions. In Fig. 5, we evaluate four sources of distribution shift. We evaluate four distribution shifts: (i) `Object Properties`, varying mass up to $4\times$ and scale up to $1.2\times$; (ii) `Initial-State Distribution`, expanding goal-position ranges from ± 1 to ± 2cm for `Push-T` and ± 5 to ± 10cm for `Peg-Insertion`; (iii) `Viewpoint`, applying 24-pixel translations and $0.85\times$ field-of-view scaling; and (iv) `Force Perturbations`, applying 80N random forces to the gripper. Fig. 6 further evaluates generalization over a wider range of initial goal positions.

Results. Figure 5 summarizes the aggregate results. Under in-distribution (ID) conditions, the two methods achieve broadly similar performance, with **DRIFT** attaining an 8.6% relative improvement in average success rate over two-phase distillation. Under OOD shifts, however, two-phase distillation loses approximately 42% of its ID success rate, compared with only 14% for **DRIFT**. Consequently, **DRIFT**'s relative advantage grows to 61% in OOD settings, demonstrating substantially stronger generalization beyond the training distribution.

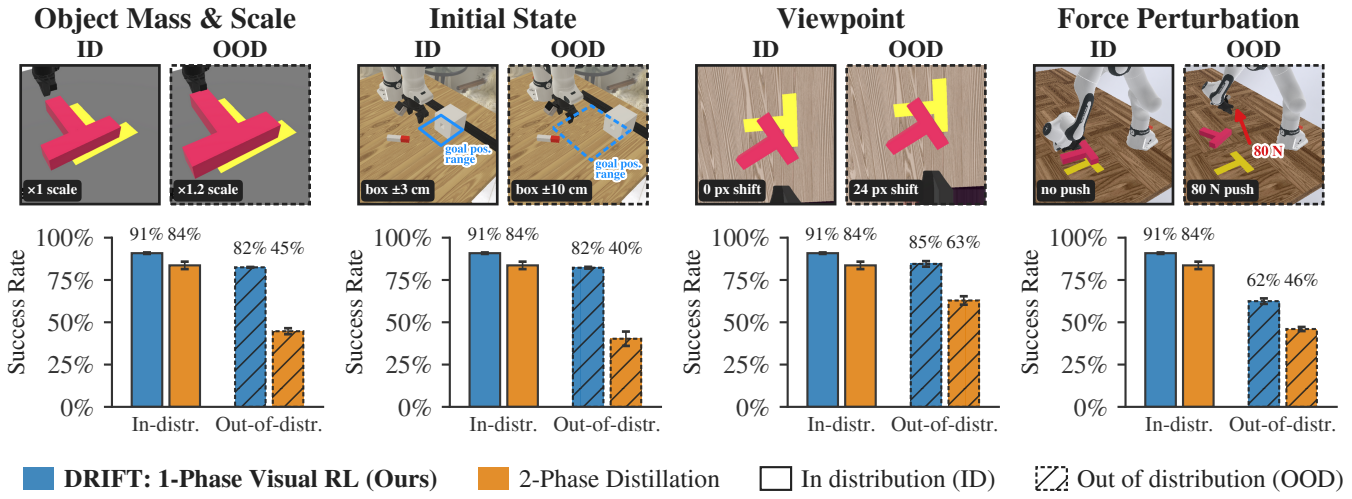

Fig. 5. **One-phase visual RL retains higher success under distribution shifts.** In-distribution (ID) and selected out-of-distribution (OOD, hatched) success for 1-Phase Visual RL and 2-Phase Distillation (DAgger with a frozen encoder). Object mass and scale use $4\times$ mass for both tasks and $1.2\times$ tee and goal scale for `Push-T`. Initial-state shifts use a 2 cm goal-position ring for `Push-T` (training jitter ± 1 cm), a 10 cm box-position ring for `Peg-Insertion` (training range ± 5 cm), and peg-start offsets in $(1, 1.25]$ times the training half-range; rings use the maximum absolute axis offset. Viewpoint shifts use a 24 px image translation for `Push-T` and $0.85\times$ camera field of view for `Peg-Insertion`. Force perturbations are `Push-T` only: 0.2 s pushes of 80 N on the gripper or 3 N on the tee. Success is episode-weighted within each selected sweep, then averaged equally across sweeps within each task and across tasks where available. Bars show means over three paired training seeds; error bars show ± 1 standard error of the mean. Top-row images are illustrative examples.

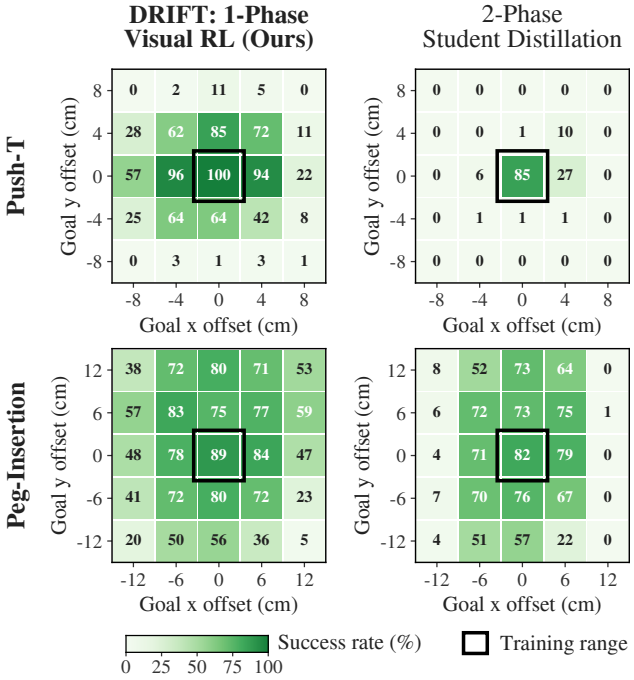

Fig. 6. **One-phase RL generalizes over a wide range of goals.** Simulation success for `Push-T` goal positions (top) and `Peg-Insertion` box positions (bottom), comparing 1-Phase Visual RL (left) with 2-Phase Student Distillation using DAgger with a frozen encoder (right). Each panel subsamples a 5×5 grid of 2 cm evaluation cells at the labeled offsets; numbers show mean success (%) over three training seeds and 100 episodes per seed and cell. Black squares mark the displayed offsets within the training range (± 1 cm for `Push-T`; ± 5 cm for `Peg-Insertion`), represented by the center cell in each panel.

We next examine goal-position generalization in greater detail in Fig. 6. Within the range of goal poses encountered during training, the two methods again perform comparably. Outside this training distribution, however, the performance of two-phase distillation degrades much more rapidly, while **DRIFT** retains high success over a broader region.

Together, these results suggest that directly optimizing the visual policy with RL produces policies that are substantially more robust to distribution shift than those obtained through two-phase privileged-policy distillation. This improved OOD robustness may also be important for sim-to-real transfer, where discrepancies between simulation and the real world inevitably induce distribution shift at deployment.

B. Zero-Shot Sim-to-Real Transfer Results

Setup. We evaluate zero-shot sim-to-real transfer across the three tasks in Section V-A, deploying policies trained entirely in simulation without real-world data or real-world gradient updates. We compare **DRIFT** against two-phase distillation with either a frozen or unfrozen visual encoder, using matched architectures and compute budgets. For each method, we select checkpoints according to Section V-C and report real-world success rate and throughput alongside the corresponding simulation success in Table 1.

Results. Despite only modest differences in simulation, various methods exhibit substantially different real-world performance. **DRIFT** achieves slightly higher simulation success across all three tasks, but its advantage becomes much larger after transfer. On `Peg-Insertion`, **DRIFT** achieves 66.7% real-world success compared with 23.3% for two-phase distillation with a frozen ViT, while reducing the sim-to-real drop from 55.2 to 15.7 percentage points. Similarly, on `Push-T`, **DRIFT** reaches 76.7% real-world success compared with 30.0% for the frozen-ViT baseline, corresponding to sim-to-real drops of 22.6 and 63.6 percentage points, respectively. Unfreezing the visual encoder during distillation further degrades transfer on both tasks. This demonstrates that similar simulation performance can

Real-world transfer across manipulation tasks.

Task	Method	Real-world performance		Simulation-to-real transfer		
		Success ↑ (%)	Throughput ↑ (successes/min)	Success in sim ↑ (%)	Sim2Real Drop ↓ (percentage points)	Real trials
Peg insertion	**DRIFT: 1-phase visual RL (ours)**	**66.7**	**4.31**	**82.3**	**15.7**	**30**
	2-phase distillation (w/ frozen ViT)	23.3	1.13	78.5	55.2	30
	2-phase distillation (w/ unfrozen ViT)	6.7	0.28	74.6	67.9	30
Pick-Cube	**DRIFT: 1-phase visual RL (ours)**	**100.0**	**28.47**	**99.7**	**−0.3**	**20**
	2-phase distillation (w/ frozen ViT)	100.0	28.74	96.4	−3.6	20
Push-T	**DRIFT: 1-phase visual RL (ours)**	**76.7**	**5.54**	**99.3**	**22.6**	**30**
	2-phase distillation (w/ frozen ViT)	30.0	1.46	93.6	63.6	30
	2-phase distillation (w/ unfrozen ViT)	16.7	0.75	92.1	75.4	30

Table 1: Zero-Shot Sim-to-Real Transfer. DRIFT achieves slightly higher success in simulation, but substantially better real-world performance across all three tasks, resulting in markedly smaller sim-to-real degradation than two-phase distillation. For each method, we report the seed with the highest real-world success. Simulation success is evaluated over 1,024 episodes per checkpoint, and throughput measures successful task completions per minute of rollout time.

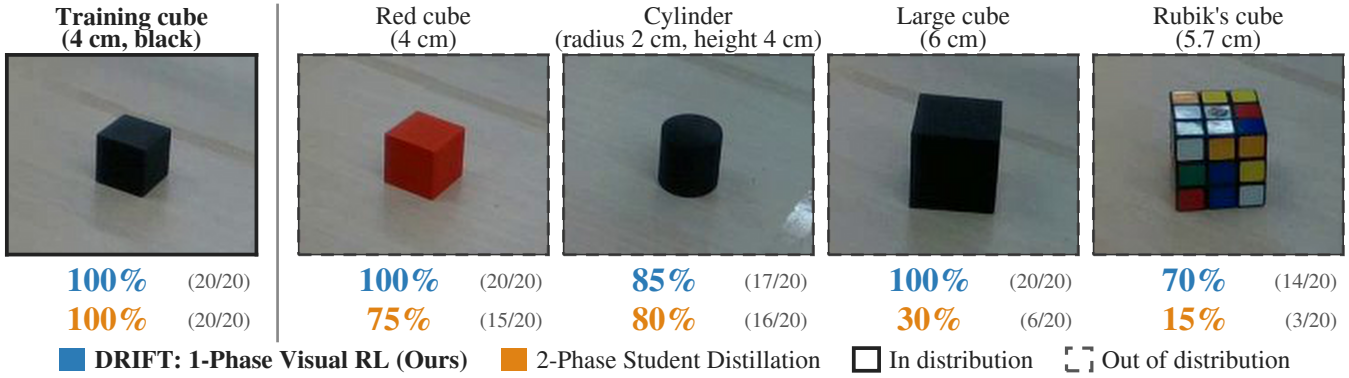

Fig. 7. **One-phase visual RL transfers better to unseen real-world objects than two-phase distillation.** Policies are trained in simulation on a single black 4,cm cube and evaluated zero-shot on four unseen objects over 20 real-world trials each. Both methods achieve 100% success on the training object, while **DRIFT** retains substantially higher success on objects with novel appearance and geometry.

conceal substantial differences in robustness to deployment shifts, with direct visual RL transferring more reliably than two-phase distillation.

On the simpler `Pick-Cube` task, both approaches transfer successfully, achieving 100% real-world success. However, this parity does not persist when evaluating the policy on unseen objects (Fig. 7). Across four unseen objects with 20 real-world trials each, **DRIFT** achieves an average success rate of 88.8%, compared with 50.0% for two-phase distillation. Thus, even on this simpler task, direct RL exhibits stronger robustness to changes in object appearance and geometry.

These results suggest that errors induced by the sim-to-real gap more readily cause failure in tasks requiring precise or extended interaction. Even on simpler tasks, increasing the deployment shift through unseen objects reveals the same robustness advantage for direct RL.

To examine these transfer differences, we compare real-world policy latents with their respective simulation rollout distributions in Fig. 8, motivated by likelihood-based typicality analysis [38]. We fit a separate Gaussian mixture model (GMM) to each policy's transformer latents collected during simulation rollouts. We evaluate the log-likelihood of real-world latents under the corresponding estimated simulation density and standardize log-likelihoods using each policy's mean and standard deviation across completed held-out simulation rollouts. Under the same real-world evaluation conditions, **DRIFT** exhibits a smaller standardized likelihood deficit than two-phase distillation during execution. This is consistent with direct RL learning more robust behavior, with its real-world latent representations remaining more typical of its simulated rollout distribution.

Real-world deployment introduces shifts in visual observations and physical dynamics that persist even after real-to-sim alignment. These shifts stress the generalization capabilities of the policy, in a similar way as the controlled OOD conditions in simulation (Section VI-A). **DRIFT** outperforms two-phase distillation by similar relative margins in both settings: approximately 60% higher success under OOD simulation conditions and 58.8% higher mean real-world success. This suggests that its stronger sim-to-real transfer stems in part from the same robustness to distribution shift observed in simulation.

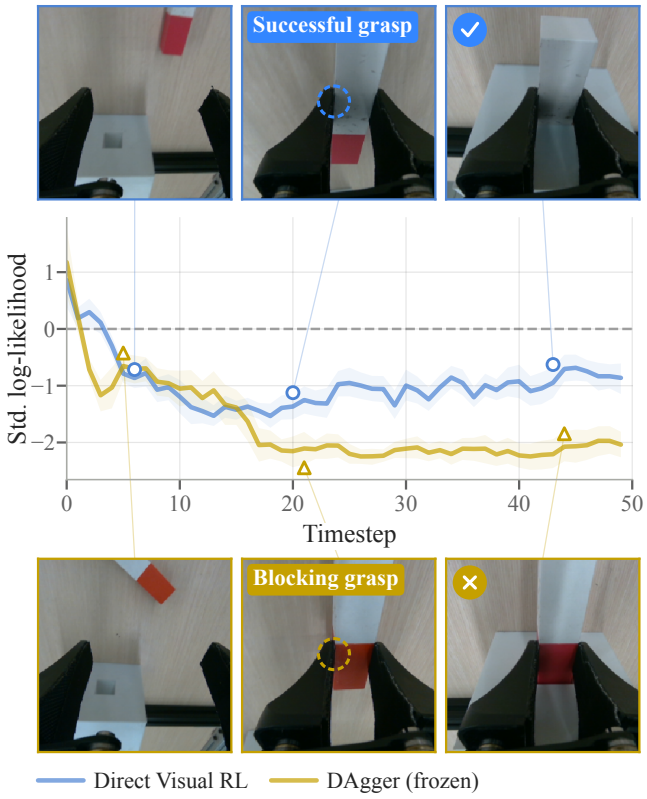

Fig. 8. **DRIFT real-world latents remain more consistent with simulation.** Mean standardized latent log-likelihood over 30 real-world episodes per policy. Scores are standardized using each policy's mean and standard deviation across completed held-out simulation rollouts. Shading denotes pointwise 95% bootstrap confidence intervals over real-world episodes. **DRIFT** (blue) exhibits a smaller standardized likelihood deficit than two-phase distillation (gold), consistent with its real-world latents remaining closer to its simulation distribution throughout rollout.

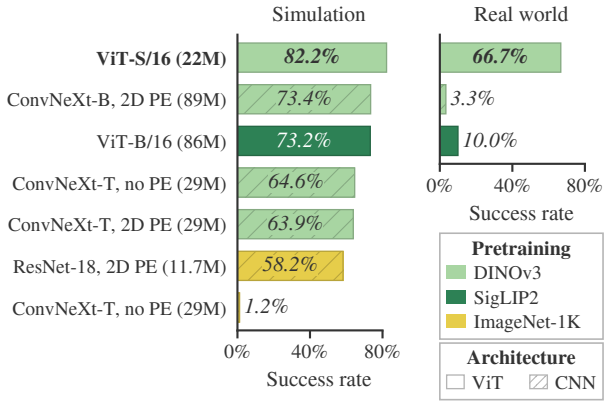

Fig. 9. **Not all frozen visual representations are equal for transfer.** Frozen visual encoder ablation on `Peg-Insertion` for seven selected encoders in simulation (left), and for the best three encoders in the real world (right). DINOv3-S/16 outperforms all other representations by a small margin in simulation, and by a significant margin in the real world.

C. Which Vision Encoder Transfers Best to the Real-world?

Setup. We compare seven visual encoder configurations on `Peg-Insertion`, spanning ViT and CNN backbones with DINOv3, SigLIP2, and ImageNet-1K pretraining. All encoders remain frozen during visual RL training. We report the simulation success rate across 512 evaluation episodes per encoder. We also evaluate sim-to-real transferability with 30 real world trials for the three best performing encoders (highest simulation success rate).

Results. Among the configurations shown, DINOv3 ViT-S/16 achieves the highest simulation success at 82.2% and the strongest real-world performance at 66.7% (Fig. 9). DINOv3 ConvNeXt-B with fixed sinuisoidal positional encodings added to the reshaped feature map and SigLIP2 ViT-B/16 also attain substantial simulation success, at 73.4% and 73.2%, respectively. However, their real-world success falls to just 3.3% and 10.0%. Despite their larger model sizes, these encoders transfer substantially worse than the 22M-parameter DINOv3 ViT-S/16. These results motivate the choice of DINOv3 ViT-S/16 for this setting, and suggest that strong simulation performance alone does not ensure effective real-world transfer.

VII. Conclusion and Remarks

We present DRIFT, a simple recipe for efficient one-phase visual RL that combines frozen pretrained vision encoders, learned attention pooling, and fast rasterized simulation. These choices address the computational burden of visual learning while preserving the spatial detail that precise manipulation demands and the robustness that real-world transfer requires. By making such representations inexpensive to use, we free more of the GPU budget for experience collection, making it practical to learn transferable manipulation policies directly from pixels.

Beyond training efficiency, DRIFT transfers substantially better than the standard two-phase distillation pipeline, achieving a 58.8% relative improvement in average real-world success across three manipulation tasks under matched architectures, observations, and compute budgets. Although the two approaches attain broadly similar performance within the simulation training distribution, DRIFT retains higher success under controlled distribution shifts and on unseen real-world objects. These results suggest that directly optimizing the visual policy with RL learns more robust behavior, helping explain why its advantage becomes substantially larger when policies encounter the discrepancies between simulation and the real world.

APPENDIX

A. Camera Calibration

We initialize simulated cameras from calibrated real-world intrinsics and camera poses estimated relative to the robot through hand–eye calibration. At deployment, images are center-cropped to a square and resized to the policy resolution. We incorporate this image-coordinate transformation into the nominal simulated intrinsics as $K_{\text{sim}} = AK_{\text{real}}$, where A is the crop-and-resize transform.[1] Together with matched extrinsics, this aligns viewing rays and projected

[1]With focal lengths $\mathbf{f} = (f_x, f_y)$, principal point $\mathbf{c} = (c_x, c_y)$, crop offset $\mathbf{o}$, and resize factor s, A gives $\mathbf{f}' = s\mathbf{f}$ and $\mathbf{c}' = s(\mathbf{c} - \mathbf{o})$ for K_{sim}.

object scales between simulation and preprocessed real images under the pinhole camera model.

B. Actuator System Identification

We record joint positions and velocities under sinusoidal chirp commands on the real robot and replay the same position targets in simulation. Using CMA-ES [39], we jointly fit PD gains, armature, and Coulomb friction coefficients for the seven arm joints by minimizing the sum of position and velocity mean-squared errors between simulated and recorded trajectories, averaged over time and joints. Each rollout starts from the recorded initial joint positions with zero velocity; torque and torque-rate limits remain fixed during fitting. To model the FR3's torque-rate constraint $|\dot{\tau}| \leq \dot{\tau}_{\max}$, we rate-limit the PD torque command at each physics step k. For each joint, with step size Δt and $\delta = \dot{\tau}_{\max}\Delta t$, we compute:

$$\tau_k^{\mathrm{PD}} = K_p(q_k^d - q_k) - K_d\dot{q}_k, \tag{4}$$

$$\tau_k = \tau_{k-1} + \mathrm{clip}(\tau_k^{\mathrm{PD}} - \tau_{k-1}, -\delta, \delta), \tag{5}$$

$$\tilde{q}_k^d = q_k + K_p^{-1}\left(\tau_k + K_d\dot{q}_k\right). \tag{6}$$

Here τ_{k-1} is the previous rate-limited torque command. To enforce torque-rate constraints, the adjusted position targets $\tilde{q}_k^{\mathrm{des}}$ are passed to the simulator's implicit joint drive, using $\dot{\tau}_{\max} = 990\,\mathrm{N\,m\,s^{-1}}$. The physics engine enforces the joint torque limits.

References

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[2] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.

[3] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.

[4] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Real-world humanoid locomotion with reinforcement learning," *Science Robotics*, vol. 9, no. 89, p. eadi9579, 2024.

[5] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[6] B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. Narang, "Industreal: Transferring contact-rich assembly tasks from simulation to reality," *arXiv preprint arXiv:2305.17110*, 2023.

[7] K. Kedia, T. G. W. Lum, J. Bohg, and C. K. Liu, "Simtoolreal: An object-centric policy for zero-shot dexterous tool manipulation," *arXiv preprint arXiv:2602.16863*, 2026.

[8] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand reorientation of novel and complex object shapes," *Science Robotics*, vol. 8, no. 84, p. eadc9244, 2023.

[9] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal, "From imitation to refinement-residual rl for precise assembly," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 01–08.

[10] I. Shenfeld, J. Pari, and P. Agrawal, "Rl's razor: Why online reinforcement learning forgets less," in *International Conference on Learning Representations*, vol. 2026, 2026, pp. 59 839–59 864.

[11] P. Yin, T. Westenbroek, Z. Zhang, I. Dagnino, E. Shilamkar, N. Mbiziwo-Tiapo, S. Bagaria, X. Liu, G. Mullins, A. Kolobov *et al.*, "Emergent dexterity via diverse resets and large-scale reinforcement learning," in *International Conference on Learning Representations*, vol. 2026, 2026, pp. 88 343–88 356.

[12] L. Chen, K. Lee, A. Srinivas, and P. Abbeel, "Improving computational efficiency in visual reinforcement learning via stored embeddings," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26 779–26 791, 2021.

[13] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa *et al.*, "Dinov3," *arXiv preprint arXiv:2508.10104*, 2025.

[14] R. Singh, A. Allshire, A. Handa, N. Ratliff, and K. Van Wyk, "Dextrahrgb: Visuomotor policies to grasp anything with dexterous hands," *arXiv preprint arXiv:2412.01791*, 2024.

[15] M. Bauza, J. E. Chen, V. Dalibard, N. Gileadi, R. Hafner, M. F. Martins, J. Moore, R. Pevceviciute, A. Laurens, D. Rao *et al.*, "Demostart: Demonstration-led auto-curriculum applied to sim-to-real with multi-fingered robots," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 6756–6763.

[16] T. He, Z. Wang, H. Xue, Q. Ben, Z. Luo, W. Xiao, Y. Yuan, X. Da, F. Castañeda, S. Sastry *et al.*, "Viral: Visual sim-to-real at scale for humanoid loco-manipulation," *arXiv preprint arXiv:2511.15200*, 2025.

[17] Y. Kim, N. Chin, A. Vasudev, and S. Choudhury, "Distilling realizable students from unrealizable teachers," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 8103–8110.

[18] N. Messikommer, J. Xing, E. Aljalbout, and D. Scaramuzza, "Student-informed teacher training," 2025. [Online]. Available: https://arxiv.org/abs/2412.09149

[19] R. Singh, K. Van Wyk, P. Abbeel, J. Malik, N. Ratliff, and A. Handa, "End-to-end rl improves dexterous grasping policies," *arXiv preprint arXiv:2509.16434*, 2025.

[20] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs *et al.*, "Mujoco playground," *arXiv preprint arXiv:2502.08844*, 2025.

[21] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T. Chan *et al.*, "Demonstrating gpu parallelized robot simulation and rendering for generalizable embodied ai with maniskill3," *Proceedings of Robotics: Science and Systems,(LosAngeles, CA, USA)*, 2025.

[22] A. Almuzairee and H. I. Christensen, "Squint: Fast visual reinforcement learning for sim-to-real robotics," *arXiv preprint arXiv:2602.21203*, 2026.

[23] R. Shah and V. Kumar, "Rrl: Resnet as representation for reinforcement learning," *arXiv preprint arXiv:2107.03380*, 2021.

[24] Z. Yuan, Z. Xue, B. Yuan, X. Wang, Y. Wu, Y. Gao, and H. Xu, "Pretrained image encoder for generalizable visual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 13 022–13 037, 2022.

[25] K.-H. Zeng, Z. Zhang, K. Ehsani, R. Hendrix, J. Salvador, A. Herrasti, R. Girshick, A. Kembhavi, and L. Weihs, "Poliformer: Scaling on-policy rl with transformers results in masterful navigators," *arXiv preprint arXiv:2406.20083*, 2024.

[26] T. Chu, Y. Zhai, J. Yang, S. Tong, S. Xie, D. Schuurmans, Q. V. Le, S. Levine, and Y. Ma, "Sft memorizes, rl generalizes: A comparative study of foundation model post-training," 2025. [Online]. Available: https://arxiv.org/abs/2501.17161

[27] J. Liu, F. Gao, B. Wei, X. Chen, Q. Liao, Y. Wu, C. Yu, and Y. Wang, "What can RL bring to VLA generalization? an empirical study," in *Advances in Neural Information Processing Systems*, 2025. [Online]. Available: https://arxiv.org/abs/2505.19789

[28] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011. [Online]. Available: https://arxiv.org/abs/1011.0686

[29] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *RSS*, 2018.

[30] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[31] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[33] G. Zhou, Z. J. Cui, A. Langford, B. Tan, Y. LeCun, and L. Pinto, "Patch policy: Efficient embodied control via dense visual representations," *arXiv preprint arXiv:2607.18236*, 2026.

[34] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira, "Perceiver: General perception with iterative attention," in *International conference on machine learning*. PMLR, 2021, pp. 4651–4664.

[35] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, "Scaling vision transformers," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2022, pp. 1204–1213.

[36] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds *et al.*, "Flamingo: a visual language model for few-shot learning," *Advances in neural information processing systems*, vol. 35, pp. 23 716–23 736, 2022.

[37] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T. kai Chan *et al.*, "Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai," in *Robotics: Science and Systems*, vol. 5, 2025.

[38] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan, "Detecting out-of-distribution inputs to deep generative models using typicality," *arXiv preprint arXiv:1906.02994*, 2019.

[39] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.