# Robot-Aligned 3D Gaussian Splatting for Precise Reaching with Wrist-Camera VLA Policies

Zuoxu Wang, Xiao Liang*

arXiv:2610.35626v1 [cs.RO] 26 Oct 2026

***Abstract*—Efficient robotic data collection remains a major bottleneck for vision-language-action (VLA) policy learning, especially for tasks requiring accurate end-effector positioning and camera-action alignment. This challenge is particularly severe in electronic-waste disassembly, where robots operate in cluttered assemblies of densely packed, visually similar components. Full digital-twin simulation is difficult because many parts are thin, reflective, or mechanically complex, making accurate object-level meshes and contact models difficult to obtain. In this work, we focus on pre-contact reaching and propose a 3D Gaussian Splatting (3DGS)-based data generation pipeline for wrist-camera VLA adaptation. Our method reconstructs the real workspace with 3DGS and renders diverse wrist-camera observations from robot-aligned poses. These observations are paired with smooth, structured reaching trajectories to provide robot-aligned visual-action supervision for VLA fine-tuning. We further augment the VLA policy with an instruction-conditioned reach verifier that estimates target proximity and provides reliable termination before physical interaction. Experiments in motherboard disassembly demonstrate precise VLA-based reaching using wrist-camera observations alone, without an external side-view camera. With $\pi$0.5, 1,500 generated trajectories achieve 80.0% precise-reaching success across three targets, compared with 58.7% using 150 teleoperated demonstrations, while requiring less than 10 minutes of active human capture. Project website: https://001-wang.github.io/PrecisionReach-GS/**

***Index Terms*—Data-efficient manipulation, robot learning, robotic data generation, vision-language-action models, 3D Gaussian splatting.**

## I. Introduction

PRECISE reaching is a fundamental requirement for real-world robotic manipulation. Before a robot can grasp, pull, insert, place, cut, or otherwise interact with an object, it must first move its end-effector to the correct target region [1]–[4]. This requirement becomes especially challenging in robotic disassembly, where targets include small components such as screws and memory modules as well as larger but fragile components such as GPUs [5], [6]. These components are often densely arranged and surrounded by neighboring parts, leaving little tolerance for reaching errors [7], [8]. Once the robot reaches the correct interaction region, subsequent actions can often be handled by task-specific controllers or skill modules, making target-oriented reaching a critical bottleneck between visual perception and successful manipulation [9].

Zuoxu Wang, Xiao Liang are with the Zachry Department of Civil and Environmental Engineering, Texas A&M University, College Station, TX 77843 USA(e-mail: zuoxu@tamu.edu; xliang@tamu.edu).

* Corresponding Author.

This work was supported by the USA National Science Foundation under Grant No.2422826. Portions of this research were conducted with the advanced computing resources provided by Texas A&M High Performance Research Computing

Vision-language-action (VLA) models provide a promising framework for learning such reaching behavior directly from visual observations and task instructions [10]–[13]. By mapping camera observations and language commands to robot actions, VLA policies can learn visually grounded motion generation without requiring each target or task to be manually encoded. With sufficiently large, diverse, and well-aligned demonstrations, VLA models have the potential to generate accurate and adaptive reaching actions across different target objects, viewpoints, and instructions. This makes VLA particularly attractive for manipulation tasks where the robot must understand both the semantic meaning of the target and its spatial location in the scene. Importantly, in many manipulation tasks, the pre-contact reaching stage occupies a large portion of the VLA-controlled action sequence, as the robot must progressively move from an initial pose to the target interaction region before executing the final physical skill. Therefore, improving the data efficiency and reliability of reaching behavior can substantially benefit the overall VLA manipulation pipeline.

However, the effectiveness of VLA models is strongly tied to data scale and data quality [14]–[17]. To learn precise reaching behavior, the policy requires many demonstrations in which the visual observation, language instruction, robot state, and action are consistently aligned. In practice, collecting such data on real robots remains difficult. Most demonstrations are collected through human teleoperation, which is slow, expensive, and difficult to scale [18], [19]. This problem becomes more severe when robot access is limited, because data collection occupies the physical robot for long periods and reduces the time available for training, testing, and other experiments. Moreover, human-collected demonstrations often introduce noise that can directly affect VLA learning. Teleoperated trajectories may be jerky, inaccurate, or contain unnecessary corrective motions [20], [21]. For example, an operator may move the robot toward an approximate region, realize that the target has been missed, and then perform several small adjustments before reaching the desired position. While this behavior is natural during manual operation, it creates ambiguous supervision for the policy: the model may learn both the intended reaching motion and the accidental correction behavior. In addition, delays between camera capture, robot state recording, and action execution can cause visual observations to be paired with slightly mismatched actions. Such camera-action misalignment is especially harmful for precise reaching, where small spatial errors can determine whether the robot reaches the correct component or misses it.

A natural alternative is to generate demonstrations in simu-

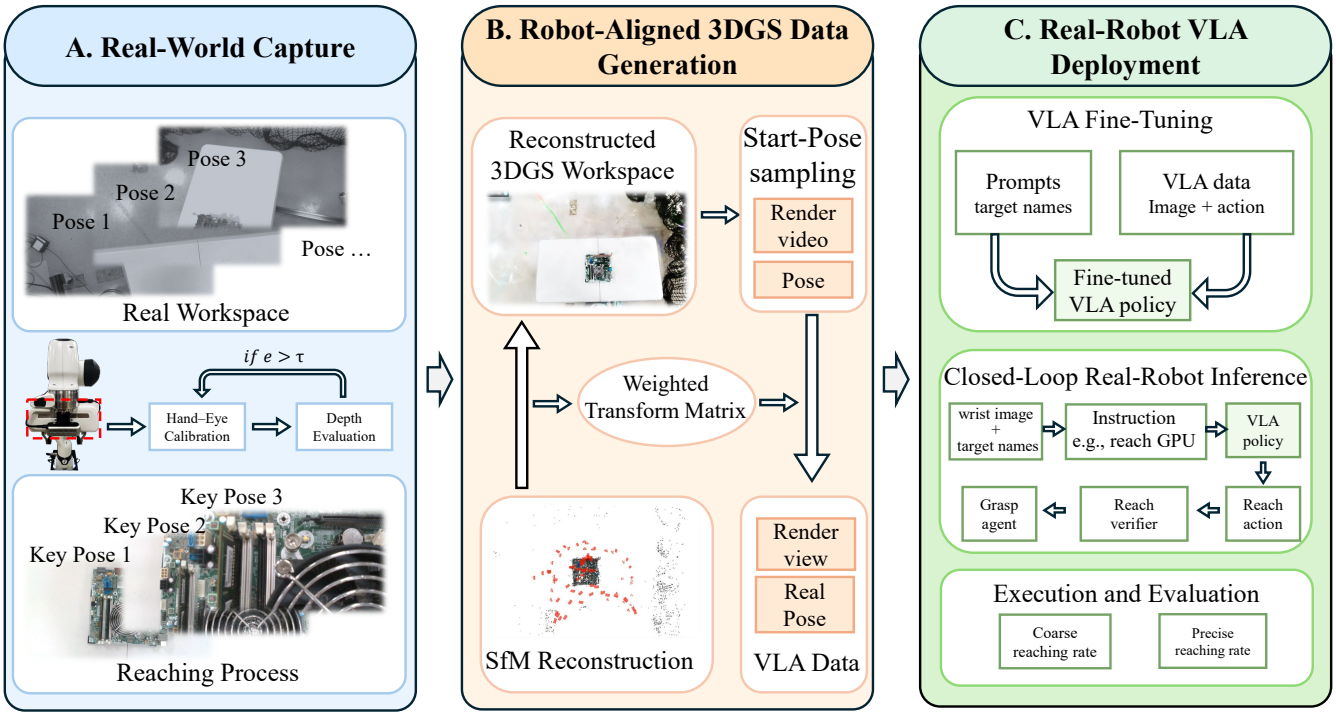



Fig. 1. Overview of the proposed real-to-sim-to-real pipeline. The system first captures and calibrates the real workspace, then reconstructs and aligns a 3DGS scene for robot-feasible wrist-camera data generation, and finally fine-tunes and deploys a VLA policy for closed-loop precise reaching.

lation, but building a full digital-twin simulator for electronic-waste disassembly is also challenging. Motherboards contain many densely packed components with different sizes, materials, and appearances, including small screws and connectors as well as larger but fragile parts such as GPUs, heat sinks, and memory modules. Many of these components are thin, reflective, or mechanically complex, making accurate object-level mesh reconstruction, material modeling, and contact simulation difficult. As a result, full simulation requires substantial manual modeling effort and can still introduce a large sim-to-real gap if the geometry, camera viewpoint, or physical interaction does not match the real workspace.

In addition to data collection and simulation, camera configuration also affects the scalability of VLA training and deployment. Many robot-learning systems use fixed external cameras or multi-view observations, which provide useful global context but require stable placement and calibration [22], [23]. In contrast, we consider a harder wrist-camera-only setting, where the viewpoint moves with the end-effector and the policy must infer both target location and motion correction from a narrow, continuously changing view. This reduces dependence on fixed side-view cameras, but also makes precise reaching more data-demanding, since the policy needs diverse and well-aligned wrist-camera observations to learn reliable corrections.

In this work, we propose *PrecisionReach-GS*, a lightweight real-to-sim-to-real pipeline for data-efficient VLA fine-tuning in a wrist-camera-only setting. As shown in Fig. 1, the pipeline first reconstructs the real task environment with 3D Gaussian Splatting (3DGS), preserving the cluttered background, object layout, and visual complexity of the physical workspace. Unlike full digital-twin simulators that require object-level meshes, collision models, and contact dynamics, PrecisionReach-GS focuses on the pre-contact reaching stage, where realistic wrist-camera observations and robot-aligned action labels are the main requirements. The system samples feasible starting poses, renders diverse wrist-camera observations, generates smooth reaching trajectories toward target components, and organizes the resulting observation–action pairs for VLA fine-tuning. During deployment, the fine-tuned VLA policy predicts reaching actions from wrist-camera observations, while a lightweight verifier determines whether the robot has reached the target interaction region. As a result, the user only specifies the target task and desired data scale, and the pipeline produces structured wrist-camera training data without large-scale teleoperation or a complete physical digital twin.

The main contributions of this work are summarized as follows:

- We propose *PrecisionReach-GS*, a lightweight real-to-sim-to-real framework that uses 3DGS as a photorealistic workspace representation for VLA adaptation, avoiding object-level meshes, contact models, and full physics-based digital twins.
- We develop a robot-aligned visual-action generation method that maps the reconstructed scene into the robot frame, samples kinematically feasible starting poses, and generates synchronized wrist-camera observations, robot states, and smooth reaching actions with minimal human intervention.
- We introduce an instruction-conditioned reach verifier

that complements the VLA policy with instruction-conditioned reach assessment, enabling more reliable stopping under the distinct geometric precision requirements of different manipulation targets.
- We experimentally validate the proposed pipeline in a robotic motherboard disassembly scenario, demonstrating that wrist-camera-only VLA fine-tuning, together with reach verification, can achieve precise target reaching and support successful downstream manipulation skills.

## II. Related work

**Robot data collection for VLA adaptation.** Vision-language-action (VLA) models have recently emerged as a promising paradigm for instruction-conditioned robot control, mapping visual observations, language instructions, and robot states directly to actions. Representative models such as RT-2, OpenVLA, OpenVLA-OFT, Octo, $\pi_0$, and $\pi_{0.5}$ show that large vision-language backbones can be adapted for visuomotor control through action tokenization, continuous action regression, action chunking, or flow-based action generation [10], [11], [13], [24]–[26]. These models are attractive for robotic disassembly because the robot must interpret component-level semantic targets while producing continuous motion toward the correct interaction region. However, adapting these models to precise, task-specific manipulation remains limited by the availability of high-quality robot action data. Existing robot data sources mainly rely on large-scale robot logs or human teleoperation [27]–[29]. Robot logs can scale with deployment but are often tied to specific embodiments and task distributions, while teleoperation provides flexible demonstrations but remains slow, labor-intensive, and noisy. This bottleneck is especially severe for component-level reaching in robotic disassembly, where small spatial errors can cause downstream manipulation to fail.

**Real-to-sim-to-real and 3DGS-based data generation for robot learning.** To reduce real-robot data requirements, recent work has explored real-to-sim-to-real pipelines that reconstruct real scenes and use simulation, rendering, or generated demonstrations for policy learning [30]. Early methods reconstruct object-level meshes from real observations and place them into simulators to generate grasp or manipulation data [31]. More recent systems extend this idea to broader policy learning, including X-SIM, which reconstructs photorealistic simulation from action-less human videos and distills privileged policies into image-conditioned policies [32]; Real2Render2Real, which generates robot training data from object scans and a single human demonstration [33]; and EmbodieDreamer, which combines physical parameter alignment with photo-realistic video generation for reinforcement and imitation learning [34].

3D Gaussian Splatting (3DGS) further supports this direction by enabling photorealistic novel-view rendering from real-world captures [35]. Recent robotic systems use 3DGS for real-to-sim reconstruction, policy learning, simulation, and evaluation. For example, RoboGSim builds a 3DGS-based Real2Sim2Real simulator with mesh-based digital twins and closed-loop policy evaluation [36]; PolaRiS reconstructs high-fidelity environments for evaluating generalist robot policies, including wrist-camera policies [37]; and GaussGym integrates 3DGS rendering into vectorized physics simulation for pixel-based locomotion learning [38]. Related systems such as SplatSim, Robo-GS, and GSWorld also explore Gaussian-based rendering and simulation for robot policy learning or evaluation [39]–[41].

TABLE I
Comparison with representative 3DGS-based robotic data generation methods.

| Method | 3DGS | Physics Sim. | Object Mesh | GPU Simulator Req. |
|---|---|---|---|---|
| SplatSim [39] | ✓ | ✓ | × | ✓ |
| RoboGSim [36] | ✓ | ✓ | ✓ | ✓ |
| Real2Render2Real [33] | ✓ | × | ✓ | ✓ |
| **PrecisionReach-GS** | ✓ | × | × | × |

These works demonstrate the value of reconstructed or generated visual environments for scalable robot learning. However, most of them target full manipulation, object-motion imitation, policy evaluation, or physics-coupled simulation, and therefore often require mesh extraction, object-level interaction modeling, dynamics simulation, or high-fidelity simulation platforms. Such pipelines introduce higher setup cost and can be challenging to build for complex scenarios.

**Positioning of our work.** Compared with prior real-to-sim-to-real and 3DGS-based robot learning systems, *PrecisionReach-GS* is not intended to build a full physical simulator or a general policy-evaluation platform. Instead, it provides a lightweight, lower-setup-cost data generation pipeline for wrist-camera VLA adaptation in pre-contact reaching. Our key distinction is that we use 3DGS as a mesh-free visual data generator aligned with the robot frame, generating feasible visual-action supervision without object-level mesh reconstruction or contact simulation. The method also differs from conventional visual servoing: it learns language-conditioned component reaching directly from wrist images instead of requiring an explicitly detected target feature and a hand-designed image-space control law.

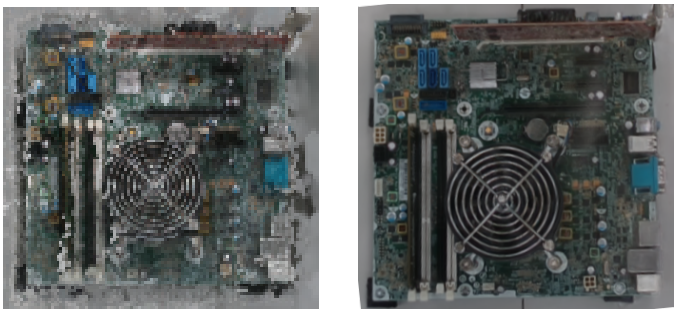
## III. Method

### A. Overview

PrecisionReach-GS consists of three stages, as illustrated in Fig. 1. First, we capture the real workspace with a wrist-mounted camera and record a small set of target-specific key poses, while hand–eye calibration establishes the metric relationship between the robot and camera frames. Second, we reconstruct the workspace with 3D Gaussian Splatting and align it to the robot base frame, enabling robot-feasible wrist-view rendering and reaching-trajectory generation. Third, the generated wrist-camera observations, target instructions, and action labels are used to fine-tune a VLA policy, which is deployed in closed loop with a reach verifier before transferring control to downstream manipulation. This design reduces the need for large-scale teleoperation while preserving the visual complexity of the real workspace.

### B. Real-Scene Reconstruction

The first stage reconstructs the real workspace in a robot-referenced form. We first perform hand–eye calibration between the robot end-effector and the wrist-mounted camera.



(a) Mesh rendering. (b) 3DGS rendering.

Fig. 2. Rendering representations reconstructed from the same workspace capture. The 3DGS representation better preserves the appearance needed for wrist-camera policy observations.

Although CAD can provide a nominal camera-to-end-effector transform, hand–eye calibration captures physical mounting offsets and remains applicable when a complete CAD-to-optical-frame chain is unavailable. Given the calibrated transformation ${}^{E}\mathbf{T}_{C}$, the camera pose in the robot base frame is

$$ {}^{B}\mathbf{T}_{C}^{(i)} = {}^{B}\mathbf{T}_{E}^{(i)}{}^{E}\mathbf{T}_{C}, \tag{1} $$

where ${}^{B}\mathbf{T}_{E}^{(i)}$ is the end-effector pose of the $i$-th observation.

Because precise pre-contact reaching is sensitive to vertical errors—that is, errors along the local surface-normal or approach direction—we validate the calibration with a depth-based plane check. Held-out observations of a planar reference surface are transformed into the robot base frame, and their point-to-plane residual is measured as

$$ e_i = \frac{\left|\mathbf{n}^{\top}\mathbf{x}_i + d\right|}{\|\mathbf{n}\|_2}, \tag{2} $$

where $(\mathbf{n}, d)$ defines the fitted reference plane and $\mathbf{x}_i$ is a newly observed 3D point. This check verifies that the wrist-camera geometry is reliable for vertical reaching, where depth errors may cause missed contact or collision with nearby components.

We then record wrist-camera videos of an unstructured workspace and reconstruct it as a photorealistic 3D Gaussian Splatting scene. The scene preserves cables, clustered components, reflective parts, and cluttered background regions without requiring object-level meshes or contact models. Short reaching videos toward target components are also recorded to obtain key poses near the desired interaction regions.

The choice of 3DGS is motivated by the observation model required for VLA training rather than by geometry alone. As illustrated in Fig. 2, a mesh reconstructed from the same capture exhibits visible geometry and appearance artifacts even when the motherboard region is retained. These artifacts directly alter the wrist-camera input to the policy. Conventional image augmentation can diversify color or crop statistics, but it cannot synthesize geometrically consistent observations along a new robot trajectory. In contrast, 3DGS preserves the appearance and surrounding context of the real scene while rendering a continuous sequence of novel views from robot-referenced poses.

### C. Robot-Aligned 3DGS Data Generation

Given the reconstructed 3DGS workspace, we generate VLA training data through robot-aligned view synthesis. Unlike free-viewpoint rendering, each synthesized wrist-camera view is derived from a robot end-effector pose in the calibrated robot base frame, ensuring that the rendered observations correspond to physically meaningful robot states.

**Robot-frame scene alignment.** The 3DGS scene is initially expressed in the SfM coordinate frame, whose scale and orientation are not directly consistent with the robot. We align the reconstruction frame $G$ with the robot base frame $B$ using matched camera centers from SfM and robot kinematics. Specifically, we estimate a weighted similarity transformation by solving

$$ \min_{s,\mathbf{R},\mathbf{t}} \sum_{i} w_i \left\| s\mathbf{R}\mathbf{c}_i^{G} + \mathbf{t} - \mathbf{c}_i^{B} \right\|_2^2, \quad \mathbf{R} \in SO(3),\ s > 0. \tag{3} $$

Here, $s$, $\mathbf{R}$, and $\mathbf{t}$ map the reconstruction frame to the robot frame. The weights $w_i$ emphasize task-relevant anchor frames near target components, improving local alignment accuracy around interaction regions.

**Feasible start-pose sampling.** To increase viewpoint and state diversity, we sample candidate starting poses around the reconstructed workspace before generating trajectories toward the target key poses. Because randomly sampled poses may be visually plausible in the 3DGS scene but infeasible for the physical robot, each candidate is filtered using conservative kinematic checks. Let $\Omega_{\text{safe}}$ denote a bounded workspace region around the task area. A candidate pose is accepted only if its position lies within this region, its inverse-kinematics solution has sufficiently small residual error, the resulting joint configuration satisfies the robot joint limits, and the configuration is not close to a kinematic singularity:

$$ \begin{aligned} \mathbf{p} &\in \Omega_{\text{safe}}, & \epsilon_{\text{IK}}(\mathbf{p}) &< \epsilon_{\max}, \\ \mathbf{q}^{-} &\le \mathbf{q}(\mathbf{p}) \le \mathbf{q}^{+}, & m(\mathbf{q}) &> m_{\min}. \end{aligned} \tag{4} $$

Here, $\epsilon_{\text{IK}}$ denotes the IK residual, $\mathbf{q}^{-}$ and $\mathbf{q}^{+}$ are the robot joint limits, and $m(\mathbf{q})$ measures the local translational manipulability of the arm. We compute

$$ m(\mathbf{q}) = \sqrt{\det\left(\mathbf{J}_p(\mathbf{q})\mathbf{J}_p(\mathbf{q})^{\top}\right)}, \tag{5} $$

where $\mathbf{J}_p$ is the translational part of the robot Jacobian. Low manipulability indicates that the robot is close to a singular configuration, where small Cartesian corrections may require large or unstable joint motions. This filtering step ensures that the generated wrist-camera observations and reaching trajectories correspond to physically executable robot states rather than only visually plausible rendered poses.

**Trajectory rendering and action labeling.** For each accepted start pose, we generate a smooth task-space trajectory from the sampled start pose to a pre-approach pose and then to the target key pose; we do not use a sampling-based collision-aware planner such as RRT. Target poses are recorded during reconstruction and serve both as interaction keyframes and as high-weight anchors in the scene-to-robot alignment. Within each segment, cubic smoothstep timing,

$$ h(u) = 3u^2 - 2u^3, \qquad u \in [0, 1], \tag{6} $$

interpolates translation and the corresponding rotation interpolation. Its zero endpoint velocity avoids the abrupt segment transitions produced by linear timing. Each intermediate end-effector pose is converted to a wrist-camera pose and rendered from the aligned 3DGS scene. The same generated trajectory is converted to the action representation required by each VLA policy. For joint-space policies such as the OpenPI-based $\pi_{0.5}$ policy, we use inverse kinematics and define the action as

$$\mathbf{q}_t = \mathrm{IK}({}^{B}\mathbf{T}_{E,t}), \quad \mathbf{a}_t^q = \mathbf{q}_{t+1} - \mathbf{q}_t. \tag{7}$$

For pose-space policies such as OpenVLA-OFT, we use the relative end-effector motion:

$$\mathbf{a}_t^x = \rho\left(({}^{B}\mathbf{T}_{E,t})^{-1}\,{}^{B}\mathbf{T}_{E,t+1}\right), \tag{8}$$

where $\rho(\cdot)$ maps the relative transform to the policy-specific pose-action format. For throughput, data-generation workers run in parallel and independently load the same aligned 3DGS scene. We apply no worker-specific or render-specific domain randomization; diversity comes from the sampled robot poses and their view-consistent trajectories.

### D. Policy Learning

The generated trajectories are converted into VLA training samples and used to adapt the reaching policy by behavior cloning. Each sample consists of a wrist-camera observation $\mathbf{I}_t$, a target instruction $\ell$, the robot state $\mathbf{s}_t$, and a reaching action $\mathbf{a}_t$. The reaching policy is then trained with the behavior-cloning objective

$$\min_{\theta} \sum_{t} \mathcal{L}_{\mathrm{BC}}\left(\pi_\theta(\mathbf{I}_t, \ell, \mathbf{s}_t), \mathbf{a}_t\right). \tag{9}$$

During deployment, the policy is executed in closed loop using the live wrist-camera image and target instruction, as illustrated in Fig. 4. To improve safety, predicted actions are scaled and clipped before execution. For joint-space policies, the commanded joint target is computed as

$$\mathbf{q}_{t+1}^{\mathrm{cmd}} = \mathrm{clip}\left(\mathbf{q}_t + \mathrm{clip}(\alpha\Delta\hat{\mathbf{q}}_t, -\delta_{\max}, \delta_{\max}), \mathbf{q}^-, \mathbf{q}^+\right), \tag{10}$$

where $\alpha$ is the action scale, $\delta_{\max}$ is the maximum joint update, and $\mathbf{q}^-, \mathbf{q}^+$ are the joint limits. For pose-space policies, the predicted relative end-effector motion is similarly scaled and clipped before being sent to the robot controller.

To balance speed and precision, we use a coarse-to-fine action chunking strategy. The policy predicts a short action horizon, and the number of actions executed per query is gradually reduced as the rollout progresses:

$$H_{k+1} \le H_k, \tag{11}$$

where $H_k$ is the executed chunk length at rollout chunk $k$. This allows fast motion at the beginning and more frequent visual feedback near the target.

### E. Instruction-Conditioned Reach Verification

Different manipulation targets impose distinct geometric requirements for successful grasping, making a universal reaching or stopping criterion inadequate. For example, a screw requires high precision along the vertical approach direction, whereas thin components such as GPUs require accurate alignment to ensure that the component lies within the gripper opening while allowing greater tolerance along other directions. Similarly, RAM modules permit greater positional variation along their longitudinal axis than across their narrow width. These target-dependent tolerances motivate an *instruction-conditioned reach verifier* that complements the VLA policy by determining whether the robot has reached a sufficiently precise interaction configuration for the instructed target.

We use a reach verifier, trained separately from the VLA backbone, to determine whether the end-effector has reached the target interaction region. The overall workflow is illustrated in Fig. 3. Its supervision is the Cartesian distance to the target, and training minimizes a Smooth-$L_1$ regression loss. At each policy query, the verifier estimates the target distance from the current wrist-camera observation, target instruction, and robot state:

$$\hat{d}_t = V(\mathbf{I}_t, \ell, \mathbf{s}_t). \tag{12}$$

The three inputs are first processed by separate visual, instruction, and state encoders. The encoded instruction conditions the visual representation through two complementary mechanisms: instruction-conditioned FiLM applies language-dependent modulation to the visual feature map, while language-guided spatial attention emphasizes image regions relevant to the instructed target. The resulting visual representation is fused with the encoded robot state for distance prediction. During training, we apply *state masking* by randomly setting the encoded state feature to zero with a probability of 0.5. This reduces reliance on workspace-specific cues such as absolute end-effector height, which can become a shortcut for target proximity under a fixed workspace, and encourages greater use of visual and instruction-conditioned cues.

At deployment, $\hat{d}_t < 0.65\,\mathrm{cm}$ is treated as a high-confidence reach. To reduce premature termination caused by transient predictions, the rollout terminates only after three consecutive high-confidence predictions, with an optional geometric check applied before termination. This persistence rule reduces premature stops and overshooting while allowing the robot to continue correcting its motion until it reaches the target region. Control can then be transferred to a downstream manipulation module.

### F. Downstream Grasp Execution

Once the reach verifier confirms that the end-effector has entered the target interaction region, control is transferred from the VLA policy to a deterministic grasp-and-place controller. At the handoff time $t^*$, the current end-effector pose is directly used as the grasp pose, without additional object-pose estimation or grasp-pose prediction. The parallel-jaw gripper is then closed with a commanded force of 70 N and a finite width tolerance to accommodate small residual alignment errors.

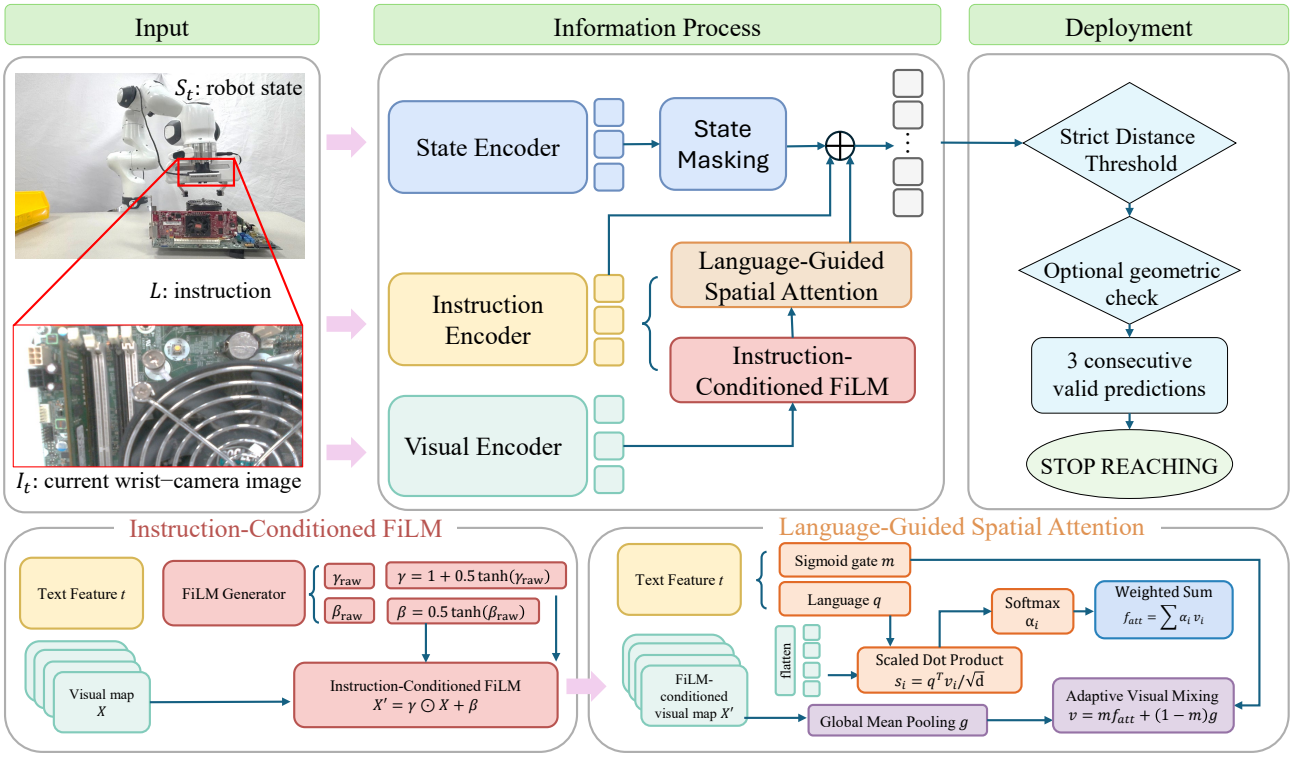



Fig. 3. Overview of the instruction-conditioned reach verifier. The verifier integrates the current wrist-camera observation, language instruction, and robot state to estimate whether the end-effector has reached the target interaction region. Language-conditioned FiLM and spatial attention adapt the visual representation to the instructed target, while state masking discourages reliance on robot-state shortcuts. During deployment, reaching is confirmed only after three consecutive valid predictions.

After a successful grasp, the robot executes a predefined lift–transfer–place sequence. The object is first lifted vertically from the grasp pose, transferred horizontally to a predefined placement region, lowered, released, and the robot then returns to its home configuration. Each Cartesian segment uses fifth-order time scaling with quaternion interpolation for orientation and is executed under joint impedance control.

The downstream controller is intentionally independent of the proposed reaching framework and is kept fixed across all trials. Its purpose is to verify that the interaction pose produced by the VLA policy and reach verifier can be directly used for physical manipulation, without introducing an additional learned manipulation policy.

## IV. Experiments and Results

### A. Experimental Setup and Evaluation Protocol

We evaluate PrecisionReach-GS on a Franka FR3 robot with a wrist-mounted RealSense D435i camera in a motherboard disassembly workspace. The dataset contains three language-conditioned targets—Screw, GPU, and RAM. The task is precise pre-contact reaching, where the robot must reach a specified component-level interaction pose before downstream manipulation. For the teleoperation baseline, an operator controls the FR3 end effector with a SpaceMouse while observing synchronized wrist- and side-camera streams. That baseline is trained and evaluated with both views, providing a stronger sensor-rich setting. In contrast, PrecisionReach-GS is trained and evaluated using only wrist-camera observations, with 300, 900, and 1,500 generated trajectories. Experiments use both OpenVLA-OFT and $\pi_{0.5}$ under the same real-robot protocol. A set of qualitative reaching trajectories is shown in Fig. 5.

We report two real-robot success metrics that isolate the learned reaching stage. Coarse reach counts arrival in the target neighborhood. Precise reach requires arrival at the target interaction region; we validate that terminal pose by executing the same downstream grasp-and-lift routine in every trial. Thus, the metric measures whether reaching supplies a manipulation-ready pose rather than asking the VLA to learn contact. The real-robot tests are conducted after physically repositioning the motherboard relative to the scene used for 3DGS reconstruction and generation. The resulting target-location changes test moderate scene variation instead of fixed-pose replay. The task geometry also makes error tolerance anisotropic: GPU and RAM targets permit somewhat more lateral error along their long axes, with additional margin from the gripper width, whereas approach-direction error remains tightly constrained for all targets.

### B. Main Real-Robot Performance and Data Efficiency

As shown in Fig. 6, the improvement is especially clear for $\pi_{0.5}$, where 900 and 1,500 PrecisionReach-GS trajectories achieve strong precise-reaching performance across all three targets. OpenVLA-OFT shows the same increasing trend but lower overall success. Notably, PrecisionReach-GS uses only

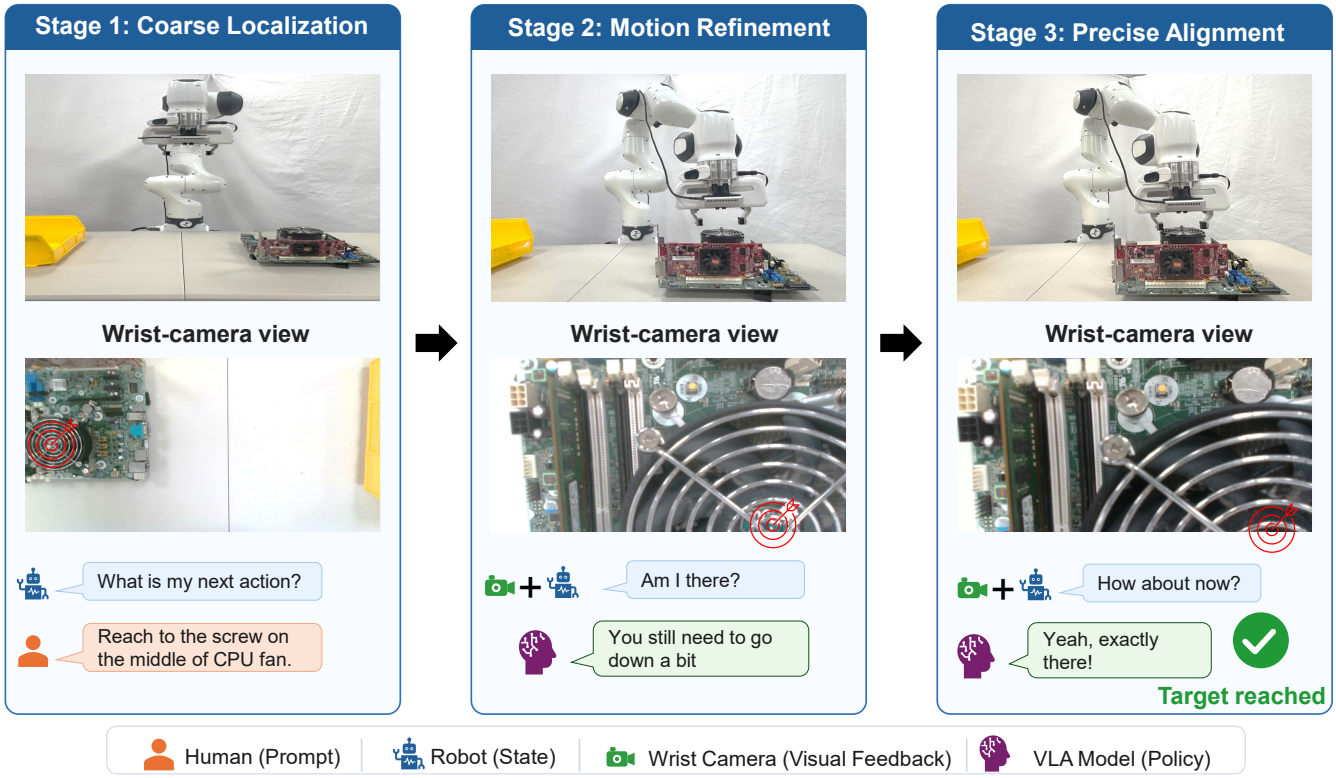



Fig. 4. Conceptual illustration of progressive wrist-camera feedback for precise reaching. The robot first performs coarse target localization, then refines its motion through repeated visual feedback, and finally stops when the reach verifier confirms alignment with the target interaction region.

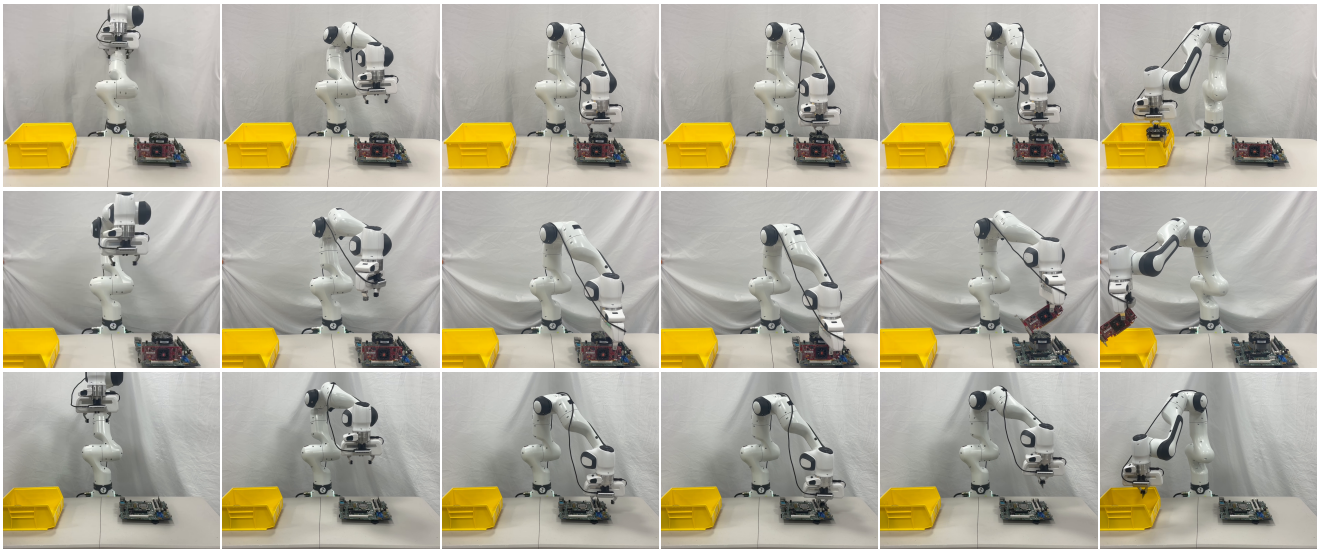


Fig. 5. Qualitative progression of three real-robot reaching trajectories (top to bottom), each shown as six frames from the initial configuration to the final target interaction pose.

the wrist camera, whereas the original teleoperation baseline uses both wrist and side views. Table II provides the corresponding per-target trial counts and makes the data-scale trend explicit.

At the largest generated-data scale, $\pi_{0.5}$ achieves 21/25, 19/25, and 20/25 precise successes for screw, GPU, and RAM, respectively (Table II). The GPU target is slightly more challenging because it is larger and taller than the other targets and requires greater wrist rotation; near the target, that rotation can occasionally cause a collision. Across both policy backbones, however, increasing the number of generated trajectories consistently improves coarse and precise reaching.

Table III compares the data-acquisition cost and trajectory characteristics of teleoperation and PrecisionReach-GS. Collecting 50–150 teleoperation demonstrations requires

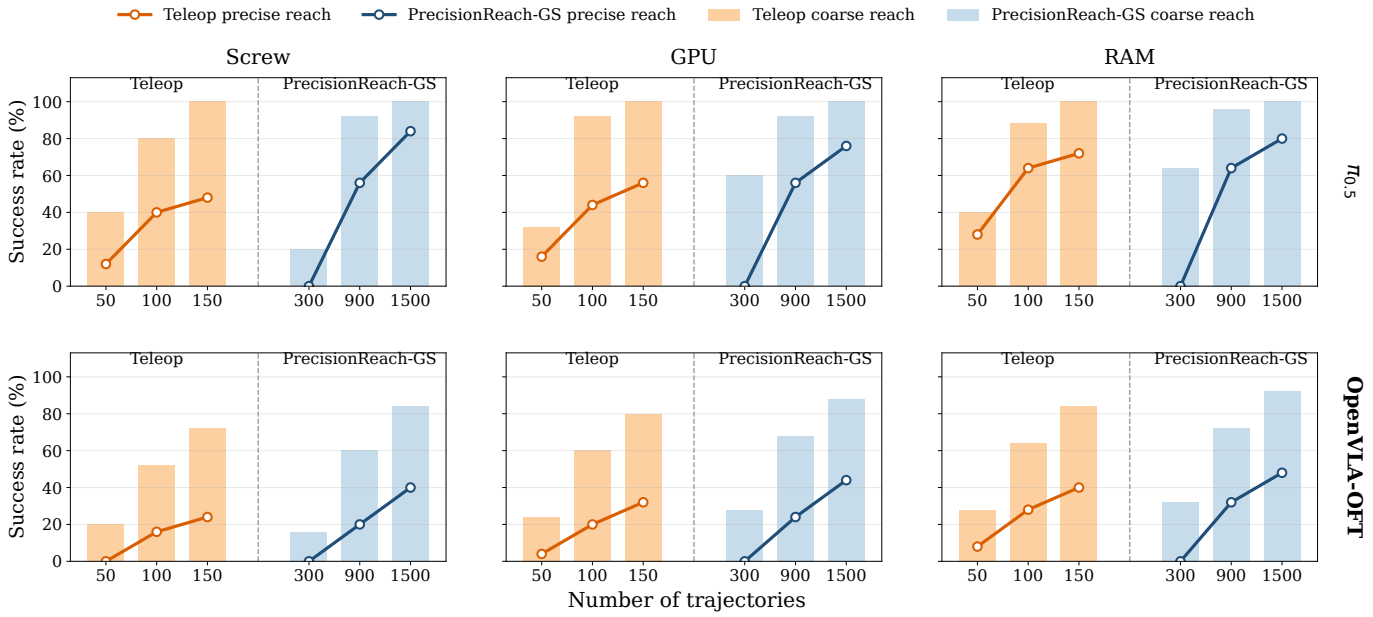



Fig. 6. Real-robot reaching performance on a physically repositioned motherboard workspace that differs from the scene used for 3DGS reconstruction and data generation. Detailed trial counts are reported in Table II.

TABLE II

REAL-ROBOT REACHING PERFORMANCE UNDER DIFFERENT AMOUNTS OF TELEOPERATION AND PRECISIONREACH-GS TRAINING DATA. EACH ENTRY REPORTS SUCCESSFUL TRIALS OUT OF 25. COARSE REACH MEASURES ARRIVAL IN THE TARGET NEIGHBORHOOD; PRECISE REACH ADDITIONALLY REQUIRES A TERMINAL POSE FROM WHICH THE FIXED GRASP-AND-LIFT ROUTINE SUCCEEDS.

| **Policy** | **Target** | **Metric** | **Teleop Trajectories** | | | **PrecisionReach-GS Trajectories** | | |
|---|---|---|---|---|---|---|---|---|
| | | | **50** | **100** | **150** | **300** | **900** | **1500** |
| $\pi_{0.5}$ | Screw | Coarse reach | 10/25 | 20/25 | 25/25 | 5/25 | 23/25 | 25/25 |
| | | Precise reach | 3/25 | 10/25 | 12/25 | 0/25 | 14/25 | 21/25 |
| | GPU | Coarse reach | 8/25 | 23/25 | 25/25 | 15/25 | 23/25 | 25/25 |
| | | Precise reach | 4/25 | 11/25 | 14/25 | 0/25 | 14/25 | 19/25 |
| | RAM | Coarse reach | 10/25 | 22/25 | 25/25 | 16/25 | 24/25 | 25/25 |
| | | Precise reach | 7/25 | 16/25 | 18/25 | 0/25 | 16/25 | 20/25 |
| OpenVLA-OFT | Screw | Coarse reach | 5/25 | 13/25 | 18/25 | 4/25 | 15/25 | 21/25 |
| | | Precise reach | 0/25 | 4/25 | 6/25 | 0/25 | 5/25 | 10/25 |
| | GPU | Coarse reach | 6/25 | 15/25 | 20/25 | 7/25 | 17/25 | 22/25 |
| | | Precise reach | 1/25 | 5/25 | 8/25 | 0/25 | 6/25 | 11/25 |
| | RAM | Coarse reach | 7/25 | 16/25 | 21/25 | 8/25 | 18/25 | 23/25 |
| | | Precise reach | 2/25 | 7/25 | 10/25 | 0/25 | 8/25 | 12/25 |

90–200 minutes of continuous human control, whereas PrecisionReach-GS only requires less than 10 minutes of active human capture to generate up to 1,500 trajectories after reconstruction. The reported generation time is measured for a single generation window; because each process uses less than 1 GB of GPU memory, multiple windows can be run in parallel depending on available GPU resources. This makes the proposed pipeline lightweight compared with simulator-based data generation pipelines that require high-memory GPU simulation environments. PrecisionReach-GS also provides smoother supervision, as reflected by its much lower RMS jerk. Table V expands this comparison beyond the mean: the generated trajectories reduce median, 95th-percentile, and maximum jerk as well, showing that the improvement is not driven by a small number of episodes.

### C. Robustness to the Initial Wrist-Camera View

We next test whether data scale improves recovery when the initial wrist image offers weak task context. Starting views are grouped into full workspace context, limited workspace context, and no workspace context. In the most challenging no-context setting, the motherboard is completely outside the initial wrist-camera field of view, although peripheral environmental cues, such as the table edge, may remain visible.

Figure 7 illustrates a representative recovery trajectory under this no-context condition. The top row shows the robot motion, while the bottom row shows the corresponding wrist-camera

TABLE III
DATA ACQUISITION COST AND TRAJECTORY SMOOTHNESS FOR TELEOPERATION AND PRECISIONREACH-GS. FOR PRECISIONREACH-GS, TOTAL TIME IS MEASURED FOR A SINGLE GENERATION WINDOW. RMS JERK IS REPORTED IN $\mathrm{m/s^3}$; LOWER IS BETTER. DETAILED JERK STATISTICS ARE REPORTED IN TABLE V.

| Method | # Traj. | Real Capture | Active Human Time | Total Time |
|---|---|---|---|---|
| Teleoperation | 50 | 50 demos | ∼90 min | ∼90 min |
| | 100 | 100 demos | ∼150 min | ∼150 min |
| | 150 | 150 demos | ∼200 min | ∼200 min |
| PrecisionReach-GS | 300 | 4 videos | $<5$ min | ∼45 min |
| | 900 | 12 videos | $<10$ min | ∼120 min |
| | 1500 | 12 videos | $<10$ min | ∼200 min |

TABLE IV
INITIAL-VIEW ROBUSTNESS UNDER DIFFERENT GENERATED DATA SCALES AND TARGET COMPONENTS. STARTING VIEWS ARE GROUPED BY THE AMOUNT OF WORKSPACE CONTEXT VISIBLE IN THE WRIST-CAMERA OBSERVATION.

| Target | Initial Visual Context | 300 Traj. | | 1500 Traj. | |
|---|---|---|---|---|---|
| | | Coarse SR | Precise SR | Coarse SR | Precise SR |
| Screw | Full workspace context | 4/10 | 0/10 | 10/10 | 10/10 |
| | Limited workspace context | 1/10 | 0/10 | 9/10 | 9/10 |
| | No workspace context | 0/10 | 0/10 | 6/10 | 2/10 |
| GPU | Full workspace context | 8/10 | 0/10 | 10/10 | 9/10 |
| | Limited workspace context | 6/10 | 0/10 | 9/10 | 9/10 |
| | No workspace context | 2/10 | 0/10 | 5/10 | 3/10 |
| RAM | Full workspace context | 8/10 | 0/10 | 10/10 | 10/10 |
| | Limited workspace context | 6/10 | 0/10 | 10/10 | 9/10 |
| | No workspace context | 3/10 | 0/10 | 6/10 | 3/10 |
| **Average** | Full workspace context | 20/30 | 0/30 | 30/30 | 29/30 |
| **Average** | Limited workspace context | 13/30 | 0/30 | 28/30 | 27/30 |
| **Average** | No workspace context | 5/30 | 0/30 | 17/30 | 8/30 |

TABLE V
TRAJECTORY SMOOTHNESS COMPARISON BETWEEN TELEOPERATION AND PRECISIONREACH-GS. RMS JERK IS REPORTED IN $\mathrm{m/s^3}$; LOWER VALUES INDICATE SMOOTHER TRAJECTORIES.

| Method | Mean ↓ | Median ↓ | P95 ↓ | Max ↓ |
|---|---|---|---|---|
| Teleoperation | 3.2657 | 3.2532 | 4.1228 | 4.5666 |
| PrecisionReach-GS | 0.2353 | 0.1683 | 0.6969 | 0.9853 |

observations. At the beginning of the trajectory, the wrist camera contains no view of the motherboard. As the robot moves, the workspace gradually enters the camera field of view, after which the policy continues toward the target region. This example qualitatively demonstrates recovery from an initial observation that contains no direct visual evidence of the task workspace.

Table IV reports the quantitative results across all three initial-view conditions and shows a large gain from 300 to 1,500 generated trajectories in every category. With 1,500 trajectories, the policy reaches 29/30 and 27/30 precise successes under full and limited context, respectively. When the workspace is initially absent, it still recovers to the target neighborhood in 17/30 trials and completes a precise reach in 8/30. Generated wrist views therefore improve tolerance to initial-view variation, although performance degrades when direct task evidence is unavailable.

### D. Controlled Comparisons and Ablations

The main comparison intentionally reflects the practical data-collection setups, but it couples data source with observation modality and trajectory smoothness. We therefore conduct two controlled $\pi_{0.5}$ experiments with 25 trials per target. First, Teleop-150 is retrained with wrist-camera input only. Second, the 900-trajectory generated dataset is regenerated with linear timing in place of the smoothstep function in (6), with every other setting fixed. Table VI shows that both the additional side view and smooth trajectory supervision contribute to performance, while the generated wrist-only setting remains directly evaluable. Data-scale effects are separately measured by Teleop-50/100/150 and PrecisionReach-GS-300/900/1500. Equal episode counts would still not equalize the quantity or quality of supervision because a teleoperated episode can contain several corrections, whereas a generated episode follows a clean structured trajectory.

We further ablate two key pipeline components using 900 generated episodes and the OpenPI implementation of $\pi_{0.5}$. Removing the verifier leaves coarse success unchanged but reduces precise success, consistent with overshoot or premature termination near the target. Removing task-keyframe weighting from the robot-frame alignment degrades both metrics (Table VII). Start-pose filtering is retained as a validity constraint: disabling workspace, IK, joint-limit, or manipulability checks produces physically infeasible labels rather than a meaningful learning baseline.

### E. Reach Verifier Ablation and Robustness

We further evaluate the reach verifier independently of the VLA policy to examine its distance-estimation accuracy, architectural components, and robustness to biased robot state. The evaluation uses 142 held-out episodes containing 21,476

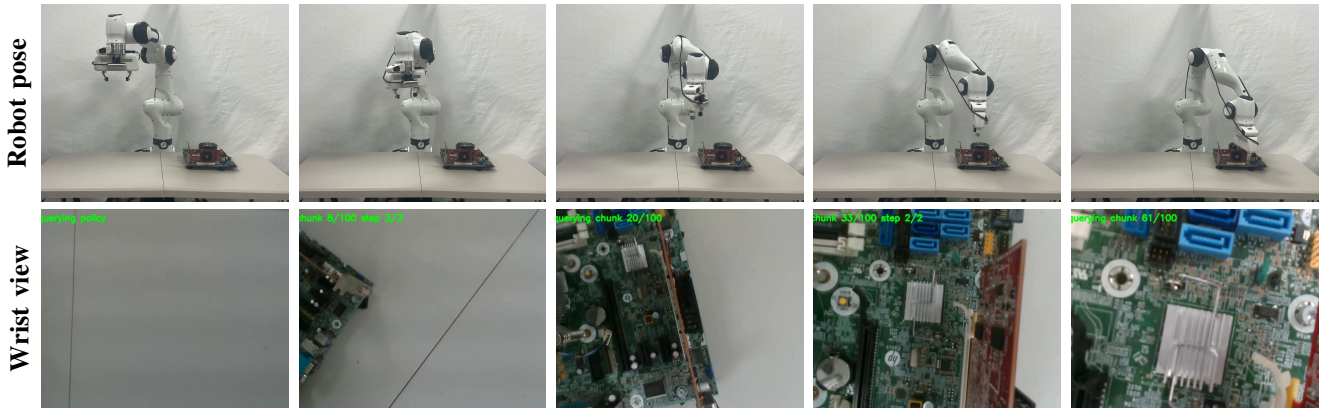



Fig. 7. Representative recovery trajectory under the no-workspace-context condition. The top row shows the robot pose and the bottom row shows the corresponding wrist-camera observation. Initially, the motherboard is completely outside the wrist camera's field of view. As the robot moves, the workspace gradually becomes visible, allowing the policy to continue toward the target region.

TABLE VI
CONTROLLED ANALYSIS WITH $\pi_{0.5}$ (25 TRIALS PER TARGET).

| Controlled factor | Original setting | Matched setting | Precise success (%) |
|---|---|---|---|
| Observation modality | Teleop-150, wrist+side | Teleop-150, wrist only | $58.7 \rightarrow 53.3$ |
| Trajectory timing | GS-900, smoothstep | GS-900, linear | $58.7 \rightarrow 50.7$ |

TABLE VII
ABLATION OF KEY PIPELINE COMPONENTS WITH 900 GENERATED EPISODES.

| Metric | Full | No verifier | No KF weighting |
|---|---|---|---|
| Coarse success (%) | 93.3 | 93.3 | 74.7 |
| Precise success (%) | 58.7 | 45.3 | 46.7 |

frames. Each learned variant is trained with three random seeds, and the best validation checkpoint from each seed is evaluated. During inference, a stop is triggered after three consecutive predicted distances below 0.65 cm. A triggered stop is considered correct when the true Cartesian distance to the instructed target is below 1 cm.

Table VIII summarizes the verifier ablations. Under clean inputs, the full model achieves an overall MAE of 0.64 cm and a near-target MAE of 0.10 cm, while maintaining 99.8% episode-level stop precision. Removing spatial attention has little effect on the overall MAE, while removing FiLM or instruction conditioning causes modest degradation. The differences become slightly more apparent near the target, where the full model achieves 0.10 cm MAE compared with 0.12 cm for the three language-conditioning ablations.

State masking introduces a clear trade-off between nominal accuracy and robustness. Without state masking, the clean-input MAE improves from 0.64 to 0.44 cm and the near-target MAE from 0.10 to 0.08 cm. However, when the end-effector height is biased downward by 2 cm, the same model fails completely, with episode-level stop precision dropping to 0%. In contrast, the full verifier retains $48.1 \pm 11.6\%$ stop precision under the same perturbation. This result indicates that state masking reduces dependence on the end-effector height, sacrificing some nominal regression accuracy in exchange for improved robustness to state bias.

Instruction conditioning also contributes to robustness under state perturbation. Removing instruction conditioning reduces biased-state stop precision from $48.1 \pm 11.6\%$ to $22.8 \pm 16.1\%$, while producing only a modest change under clean inputs. This suggests that instruction-conditioned visual features provide an additional cue when the robot-state input becomes unreliable. In contrast, the small differences between the full model and the individual FiLM and spatial-attention ablations under clean conditions indicate that neither mechanism alone dominates the verifier performance.

### F. Rendering Fidelity and Robot-Frame Alignment

The preceding experiments measure end-to-end behavior. We finally examine two representation properties that make those rollouts possible: whether rendered wrist views resemble real observations and whether reconstructed poses are accurate in the robot frame.

For visual fidelity, we compare real robot videos with videos rendered from the reconstructed scene along the corresponding camera motion. Figure 8 shows representative real and rendered frames at corresponding viewpoints. The rendered observations closely preserve the overall scene appearance, component layout, and viewpoint-dependent visual content of the real wrist-camera observations, while small local differences remain due to reconstruction and illumination variations. Table IX reports complementary image-level metrics. The SSIM reflects structural and layout consistency, correlation captures brightness and shape trends, and MSE similarity measures pixel-level agreement. The mean correlation of 0.843 indicates that the rendered views preserve major appearance trends and scene layout. The lower SSIM is consistent with the sensitivity of wrist-camera imagery to small viewpoint and illumination changes and local reconstruction artifacts.

TABLE VIII
ABLATION AND ROBUSTNESS ANALYSIS OF THE REACH VERIFIER. DISTANCE METRICS ARE EVALUATED USING THE CORRECT TARGET INSTRUCTION. NEAR-TARGET MAE IS COMPUTED OVER FRAMES WHOSE GROUND-TRUTH TARGET DISTANCE IS BELOW 3 CM. STATE ROBUSTNESS IS EVALUATED BY APPLYING A $-2$ CM BIAS TO THE END-EFFECTOR HEIGHT.

| **Verifier** | **MAE (cm)** $\downarrow$ | **Near-Target MAE (cm)** $\downarrow$ | **Stop Prec. (%)** $\uparrow$ | **Biased-State Prec. (%)** $\uparrow$ |
|---|---|---|---|---|
| Full | $0.64 \pm 0.02$ | $0.10 \pm 0.02$ | 99.8 | $48.1 \pm 11.6$ |
| w/o spatial attention | $0.64 \pm 0.07$ | $0.12 \pm 0.02$ | 99.8 | – |
| w/o FiLM | $0.67 \pm 0.06$ | $0.12 \pm 0.03$ | 99.5 | – |
| w/o instruction conditioning | $0.71 \pm 0.05$ | $0.12 \pm 0.03$ | 99.3 | $22.8 \pm 16.1$ |
| w/o state masking | $\mathbf{0.44 \pm 0.01}$ | $\mathbf{0.08 \pm 0.01}$ | **100.0** | 0.0 |

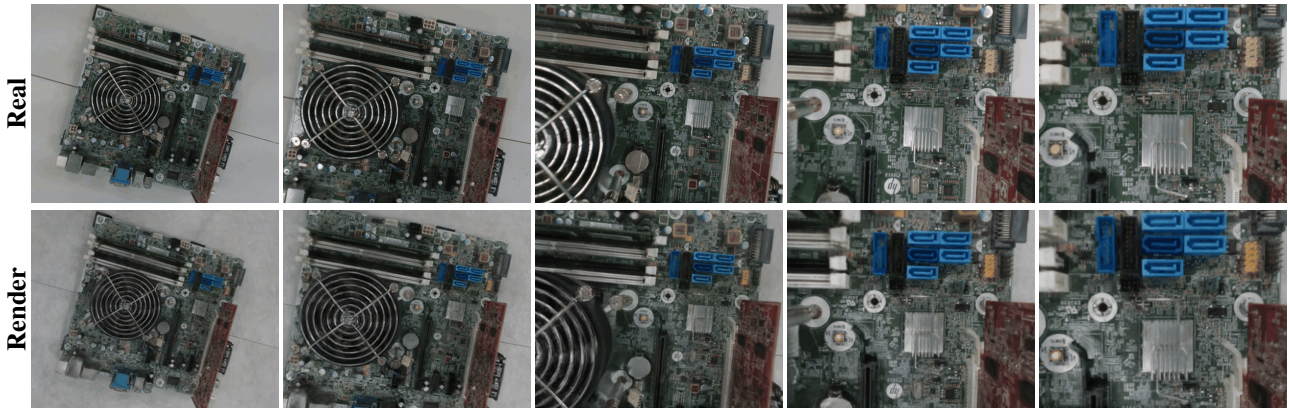



Fig. 8. Comparison between real and rendered observations.

TABLE IX
VISUAL SIMILARITY BETWEEN REAL AND RENDERED ROBOT VIDEOS.

| Metric | Mean | Best | Worst |
|---|---|---|---|
| SSIM | 0.682 | 0.919 | 0.503 |
| Correlation | 0.843 | 0.964 | 0.750 |
| MSE similarity | 0.979 | 0.991 | 0.963 |

TABLE X
ROBOT-FRAME ALIGNMENT ACCURACY AFTER TRANSFORMING THE RECONSTRUCTED SCENE INTO THE ROBOT COORDINATE FRAME.

| Metric | S1 | S2 | S3 |
|---|---|---|---|
| Position RMSE (m) | 0.0114 | 0.0094 | 0.0096 |
| Position mean error (m) | 0.0102 | 0.0077 | 0.0076 |
| Position max error (m) | 0.0258 | 0.0326 | 0.0329 |
| Anchor position mean error (m) | 0.0039 | 0.0032 | 0.0040 |
| Anchor position max error (m) | 0.0047 | 0.0042 | 0.0067 |
| Orientation mean error (deg) | 0.5792 | 0.4894 | 0.4614 |
| Orientation max error (deg) | 1.1106 | 1.7806 | 1.3140 |
| Anchor orientation mean error (deg) | 0.4573 | 0.2329 | 0.4389 |
| Anchor orientation max error (deg) | 0.5730 | 0.3624 | 0.9064 |

For geometric accuracy, we evaluate the weighted similarity transformation that maps the COLMAP/3DGS reconstruction into the robot base frame. Table X shows centimeter-level position accuracy in three scenarios, with position RMSE from 0.0094 to 0.0114 m. The task-keyframe anchors are tighter, with mean position errors below 0.0040 m in every scenario. Mean orientation errors remain below $0.58°$. Together with the end-to-end results, these measurements show that the representation is sufficiently aligned for robot-frame wrist-view rendering and feasible reaching trajectory generation.

## V. DISCUSSION AND LIMITATIONS

Repositioning the motherboard without rebuilding the dataset shows that the policy can tolerate moderate target-location changes rather than memorizing a fixed robot pose. However, this does not imply arbitrary layout generalization; larger workspace or component changes may require a new reconstruction and target demonstrations, after which the automated pipeline can regenerate training trajectories. Performance may also degrade under calibration errors, out-of-distribution viewpoints, persistent occlusion, or severe reconstruction artifacts.

The current system focuses on precise target reaching rather than contact-rich manipulation or post-grasp goal selection. Future work could combine PrecisionReach-GS for long-horizon approach with a contact-aware residual policy using visual, force, or tactile feedback for grasping and subsequent manipulation.

## VI. CONCLUSION

We presented *PrecisionReach-GS*, a lightweight real-to-sim-to-real pipeline for data-efficient VLA adaptation in precise robotic reaching. It uses a robot-aligned 3DGS workspace to generate wrist-camera observations and smooth, feasible reaching trajectories without large-scale teleoperation, object-level meshes, contact models, or a complete physics-based digital twin. Experiments on motherboard disassembly show reduced active human collection effort and improved manipulation-ready reaching over limited teleoperation data. The controlled comparisons and component ablations further identify the roles of observation modality, smooth trajectory timing, keyframe-weighted alignment, and stop verification.

These results support robot-aligned visual generation as a practical route to wrist-camera-only VLA learning in cluttered workspaces.

## REFERENCES


[1] Q. M. Marwan, S. C. Chua, and L. C. Kwek, "Comprehensive review on reaching and grasping of objects in robotics," *Robotica*, vol. 39, no. 10, pp. 1849–1882, 2021.

[2] A. Skuric, N. T. Alberto, L. Joseph, V. Padois, and D. Daney, "Online approach to near time-optimal task-space trajectory planning," *IEEE Transactions on Robotics*, vol. 42, pp. 637–652, 2026.

[3] V. W. Oguntosin, Y. Mori, H. Kim, S. J. Nasuto, S. Kawamura, and Y. Hayashi, "Design and validation of exoskeleton actuated by soft modules toward neurorehabilitation—vision-based control for precise reaching motion of upper limb," *Frontiers in neuroscience*, vol. 11, p. 352, 2017.

[4] C. Robin and S. Lacroix, "Multi-robot target detection and tracking: taxonomy and survey," *Autonomous Robots*, vol. 40, no. 4, pp. 729–760, 2016.

[5] S. Tian, X. Liang, S. Behdad, and M. Zheng, "Redefining end-of-life: Intelligent automation for electronics remanufacturing systems," *arXiv preprint arXiv:2604.03066*, 2026.

[6] G. Foo, S. Kara, and M. Pagnucco, "Challenges of robotic disassembly in practice," *Procedia CIRP*, vol. 105, pp. 513–518, 2022.

[7] C. Liu, S. Tian, X. Liang, and M. Zheng, "Self-vla: A skill enhanced agentic vision-language-action framework for contact-rich disassembly," *arXiv preprint arXiv:2603.11080*, 2026.

[8] Y. Shi, D. Wen, G. Chen, E. Welte, S. Liu, K. Peng, R. Stiefelhagen, and R. Rayyes, "Viso-grasp: vision-language informed spatial object-centric 6-dof active view planning and grasping in clutter and invisibility," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 14 931–14 938.

[9] H. T. N. Le and H. Q. T. Ngo, "Application of the vision-based deep learning technique for waste classification using the robotic manipulation system," *International Journal of Cognitive Computing in Engineering*, vol. 6, pp. 391–400, 2025.

[10] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*. PMLR, 2023, pp. 2165–2183.

[11] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[12] R. Sapkota, Y. Cao, K. I. Roumeliotis, and M. Karkee, "Vision-language-action models: Concepts, progress, applications and challenges," *arXiv preprint arXiv:2505.04769*, 2025.

[13] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai *et al.*, "$\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization," *arXiv preprint arXiv:2504.16054*, 2025.

[14] N. Pfaff, E. Fu, J. Binagia, P. Isola, and R. Tedrake, "Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025, pp. 6296–6303.

[15] S. Höfer, K. Bekris, A. Handa, J. C. Gamboa, M. Mozifian, F. Golemo, C. Atkeson, D. Fox, K. Goldberg, J. Leonard *et al.*, "Sim2real in robotics and automation: Applications and challenges," *IEEE transactions on automation science and engineering*, vol. 18, no. 2, pp. 398–400, 2021.

[16] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," *arXiv preprint arXiv:2310.17596*, 2023.

[17] P. Martinez-Gonzalez, S. Oprea, A. Garcia-Garcia, A. Jover-Alvarez, S. Orts-Escolano, and J. Garcia-Rodriguez, "Unrealrox: an extremely photorealistic virtual reality environment for robotics simulations and synthetic data generation," *Virtual Reality*, vol. 24, no. 2, pp. 271–288, 2020.

[18] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1048–1055.

[19] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE international conference on robotics and automation (ICRA)*. Ieee, 2018, pp. 5628–5635.

[20] E. Richards and L. Dai, "Modeling imitation learning robustness to noisy demonstrations via sigmoid degradation," in *2025 IEEE MIT Undergraduate Research Technology Conference (URTC)*. IEEE, 2025, pp. 1–5.

[21] L. Guo, Z. Xue, Z. Xu, and H. Xu, "Demospeedup: Accelerating visuomotor policies via entropy-guided demonstration acceleration," in *Proceedings of The 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Lim, S. Song, and H.-W. Park, Eds., vol. 305. PMLR, 27–30 Sep 2025, pp. 599–609. [Online]. Available: https://proceedings.mlr.press/v305/guo25a.html

[22] J. W. Kim, K. Wang, Z. Fu, S. Chen, J. Lai, C. Finn *et al.*, "Ego-pi: Vla fine-tuning for ego-centric human and robot data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026, pp. 1515–1524.

[23] G. Zhao, L. Guo, Y. Mei, Z. Zhu, Y. Zhang, B. Cao, M. Yu, X. He, J. Jiang, and J. Liu, "Atlasvla: Persistent world-ego state modeling for vision-language-action models," *arXiv preprint arXiv:2608.06729*, 2026.

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

[25] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu *et al.*, "Octo: An open-source generalist robot policy," *arXiv preprint arXiv:2405.12213*, 2024.

[26] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "$\pi_0$: : A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[27] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[28] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du *et al.*, "Bridgedata v2: A dataset for robot learning at scale," in *Conference on Robot Learning*. PMLR, 2023, pp. 1723–1736.

[29] H.-S. Fang, H. Fang, Z. Tang, J. Liu, C. Wang, J. Wang, H. Zhu, and C. Lu, "Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 653–660.

[30] H. Fan, H. Dai, J. Zhang, J. Li, Q. Yan, Y. Zhao, M. Gao, J. Wu, H. Tang, and H. Dong, "Twinaligner: Visual-dynamic alignment empowers physics-aware real2sim2real for robotic manipulation," *arXiv preprint arXiv:2512.19390*, 2025.

[31] L. Wang, R. Guo, Q. Vuong, Y. Qin, H. Su, and H. I. Christensen, "A real2sim2real method for robust object grasping with neural surface reconstruction," in *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, 2023.

[32] P. Dan, K. Kedia, A. Chao, E. W. Duan, M. A. Pace, W.-C. Ma, and S. Choudhury, "X-sim: Cross-embodiment learning via real-to-sim-to-real," *arXiv preprint arXiv:2505.07096*, 2025.

[33] J. Yu, L. Fu, H. Huang, K. El-Refai, R. A. Ambrus, R. Cheng, M. Z. Irshad, and K. Goldberg, "Real2render2real: Scaling robot data without dynamics simulation or robot hardware," *arXiv preprint arXiv:2505.09601*, 2025.

[34] B. Wang, X. Meng, X. Wang, Z. Zhu, A. Ye, Y. Wang, Z. Yang, C. Ni, G. Huang, and X. Wang, "Embodiedreamer: Advancing real2sim2real transfer for policy training via embodied world modeling," *arXiv preprint arXiv:2507.05198*, 2025.

[35] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–14, 2023.

[36] X. Li, J. Li, Z. Zhang, R. Zhang, F. Jia, T. Wang, H. Fan, K.-K. Tseng, and R. Wang, "Robogsim: A real2sim2real robotic gaussian splatting simulator," *arXiv preprint arXiv:2411.11839*, 2025.

[37] A. Jain, M. Zhang, K. Arora, W. Chen, M. Torne, M. Z. Irshad, S. Zakharov, Y. Wang, S. Levine, C. Finn, W.-C. Ma, D. Shah, A. Gupta, and K. Pertsch, "Polaris: Scalable real-to-sim evaluations for generalist robot policies," *arXiv preprint arXiv:2512.16881*, 2025.

[38] A. Escontrela, J. Kerr, A. Allshire, J. Frey, R. Duan, C. Sferrazza, and P. Abbeel, "Gaussgym: An open-source real-to-sim framework for learning locomotion from pixels," *arXiv preprint arXiv:2510.15352*, 2025.

[39] M. N. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal, "Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting," *arXiv preprint arXiv:2409.10161*, 2024.
[40] H. Lou, Y. Liu, Y. Pan, Y. Geng, J. Chen, W. Ma, C. Li, L. Wang, H. Feng, L. Shi, L. Luo, and Y. Shi, "Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation," *arXiv preprint arXiv:2408.14873*, 2024.
[41] G. Jiang, H. Chang, R.-Z. Qiu, Y. Liang, M. Ji, J. Zhu, Z. Dong, X. Zou, and X. Wang, "Gsworld: Closed-loop photo-realistic simulation suite for robotic manipulation," *arXiv preprint arXiv:2510.20813*, 2025.