# LAMP: LIFT IMAGE-EDITING AS GENERAL 3D PRIORS FOR OPEN-WORLD MANIPULATION

**Anonymous authors**
Paper under double-blind review




Figure 1: We propose **LAMP**, which lifts image editing as general 3D priors, enabling open-world manipulation of diverse tasks from monocular RGB-D observations and promptable instructions.

## ABSTRACT

Achieving generalization in open-world robotic manipulation remains a core challenge due to the vast diversity of task structures and environments. Existing data-driven approaches often couple spatial reasoning implicitly with robot kinematics, whereas recent LLM/VLM-based methods rely on sparse, discrete proxies (e.g., language, code, or keypoints) that struggle to express dense and continuous 3D geometric relations. To address these limitations, we propose **LAMP**, a novel framework that formulates manipulation as continuous inter-object $SE(3)$ transformations. Our central insight is that large-scale image-editing models inherently encode rich 2D spatial priors as dense visual signals, which can be robustly lifted into explicit 3D geometric foundations. Given a language instruction, LAMP synthesizes a visual goal state and extracts precise 3D transformations from single-view RGB-D observations, systematically overcoming generative scale inconsistencies and severe depth noise. Extensive experiments demonstrate that our continuous 3D formulation inherently resolves the fragility of sparse symbolic proxies, unlocking strong zero-shot generalization and versatile promptability across diverse, high-precision real-world manipulation tasks.

## 1 INTRODUCTION

Achieving open-world robotic manipulation remains a core challenge in embodied intelligence due to the vast diversity of task structures and environments. Traditional methods (Bicchi & Kumar, 2000; Mordatch et al., 2012; Rus, 1999) typically rely on task-specific analytical modeling or hand-crafted heuristics, severely limiting their scalability. Conversely, recent data-driven approaches, including reinforcement learning (RL) (Aytar et al., 2018; Rajeswaran et al., 2018; Xu et al., 2023; Hansen et al.; Moerland et al., 2023; Handa et al., 2023; Hong et al., 2025; Kumar et al., 2023; Zakka et al., 2022), imitation learning (IL) (Jarrett et al., 2020; Reuss et al., 2023; Seo et al., 2023;



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Chi et al., 2025), and VLAs (Black et al., 2025; Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2025; Li et al., 2024; Belkhale et al., 2024; O'Neill et al., 2024; Stone et al.), instead learn manipulation policies directly from robot trajectories in an end-to-end manner. However, by implicitly coupling spatial reasoning with robot-centric kinematics, these action-grounded models lack an explicit, structured representation of inter-object geometries, making them struggle to generalize across novel tasks and unseen domains. This highlights the critical need for an alternative representation: one that explicitly captures task-agnostic inter-object spatial and geometric relations.

One promising direction for open-world manipulation is to leverage the spatial reasoning ability of LLMs (Achiam et al., 2023; Team et al., 2024b;a; Bai et al., 2023) and VLMs (Liu et al., 2023; Wang et al., 2024; Bai et al., 2025; Guo et al., 2025). Some works (Liang et al., 2023; Huang et al., 2023) utilize the code-generation capability of LLMs to translate language instructions into executable manipulation primitives. While these methods achieve partial visual grounding through zero-shot perception or proxy value maps, discretizing continuous 3D actions into code semantics inherently abstracts away rich geometric details. Other methods (Huang et al., 2025; 2024; Fang et al., 2024; Nasiriany et al., 2024; Pan et al., 2025) attempt to ground manipulation more directly by extracting spatial entities (e.g., 3D keypoints, bounding boxes, or action vectors) and formulating geometric constraints. However, these sparse, entity-centric representations often reduce complex object interactions to simplified point-to-point heuristics, which are sensitive to occlusion and noisy depth estimation. Ultimately, both paradigms rely on sparse, discrete proxies (e.g., language, code, or keypoints) to approximate the continuous 3D world. They struggle to express dense geometric relations, such as contact surfaces, relative rotations, or precise full-body alignment, which are essential for high-precision manipulation like assembly (Chen et al., 2022). Video-based action models (Bharadhwaj et al., 2024; Patel et al.) suffer from visual and physical inconsistencies, hindering generalization to real robotic settings. Meanwhile, 4D generative models (Zhen et al., 2025) offer richer spatial structures but remain computationally prohibitive and lack a functional understanding of object interactions. Therefore, we seek a representation that is 1) **continuous and geometry-aware**: capable of expressing dense, fine-grained 3D spatial relations and contact geometry beyond sparse constraints, 2) **heuristic-free**: extractable directly from visual observations without predefined geometric parsing or rigid templates, and 3) **task-generalizable**: applicable across novel tasks and unseen environments without task-specific training data.

We observe that large-scale image-editing models implicitly encode rich spatial priors—dictating how objects move, rotate, and interact—as dense and continuous visual signals rather than discrete symbolic constraints. Their inherently object-preserving nature ensures that these spatial relations remain geometrically consistent across diverse scenes, enabling zero-shot generalization to novel tasks and unseen environments. Consequently, we formulate manipulation as inter-object 3D transformations: a continuous, physically grounded representation that naturally captures relative motion, contact geometry, and precise alignment. Our central insight is that the implicit 2D priors embedded in image-editing models can be lifted into explicit 3D transformations, serving as a powerful, generalizable geometric foundation for open-world manipulation.

We therefore introduce **LAMP**. Given a language instruction, LAMP synthesizes a visual goal state via image editing, lifts both the current and edited observations into 3D coordinate frames, and computes the precise inter-object $SE(3)$ transformation between them. This dense 3D transformation serves as a continuous, geometry-aware prior that naturally encodes spatial alignment and interaction intent for manipulation planning. Besides, LAMP incorporates geometry-aware point cloud filtering to mitigate depth noise, and introduces a scale-alignment mechanism to resolve object scale inconsistencies inherently introduced by 2D generative models. Our main contributions are:

- **Precise and Robust 2D-to-3D Lifting:** We propose LAMP, which lifts 2D image-editing priors into highly precise inter-object $SE(3)$ transformations. Our method systematically overcomes the inherent challenges of single-view RGB-D observations by effectively resolving 2D generative scale inconsistencies and mitigating severe depth noise under viewpoint variations.
- **Addressing Discrete Bottlenecks:** We provide an in-depth failure analysis of existing VLM/LLM-based open-world manipulation methods and demonstrate our continuous 3D formulation inherently resolves the fragility of sparse symbolic proxies, effectively unleashing the strong zero-shot generalization capabilities of VLMs for open-world robotic manipulation.
- **Extensive Task Generalization and Multimodal Promptability:** We conduct comprehensive simulation and real-world experiments across diverse tasks, including high-precision assembly, articulated object manipulation, and long-horizon tasks, demonstrating that LAMP consistently

outperforms existing baselines. Featuring versatile promptability (language, color, and object pose), LAMP provides a highly adaptable and robust framework for open-world manipulation.

## 2 Related Works

**General Representations for Manipulation.** Early works learn dense neural descriptors or 6D object poses as object-centric representations (Simeonov et al., 2022; Tremblay et al., 2018), but struggle to generalize to unseen objects. Recent VLM/LLM-based methods extract semantic keypoints (Huang et al., 2025; Nasiriany et al., 2024; Fang et al., 2024), motion vectors (Huang et al., 2024), or explicit 3D poses (Pan et al., 2025). While interpretable, these sparse, discrete proxies are fragile under occlusion and depth noise, and are fundamentally limited in expressing continuous 3D geometric relations. Alternatively, 3D flow (Eisner & Zhang, 2022; Gao et al.; Dharmarajan et al., 2025) captures motion but lacks structural grounding between interacting objects. While some works learn inter-object transformations for assembly (Zhao et al., 2025; Chen et al., 2022; Wu et al., 2023), they typically require complete 3D geometry and task-specific training. In contrast, LAMP extracts highly precise inter-object $SE(3)$ transformations from single-view RGB-D observations via 2D image-editing priors, requiring no task-specific training or manual annotations.

**Foundation Models for Manipulation.** Foundation models have been widely adopted to facilitate embodied reasoning and manipulation (Kawaharazuka et al., 2024; Firoozi et al., 2025). VLAs (Brohan et al., 2022; Black et al., 2025; Kim et al., 2025) co-fine-tune vision-language models with robot trajectories for low-level control, but their reliance on large-scale robot data limits generalization to novel tasks. To address this, video-based action models leverage video diffusion priors to capture physical dynamics: VPP (Hu et al., b) conditions robot policies on predictive representations from video diffusion models; UniPi, Vidar and Vidarc (Du et al., 2023; Feng et al., 2025a;b) combine video diffusion with masked inverse dynamics. World action models such as DreamZero (Ye et al.) go further by jointly predicting future world states and actions, using video as a dense representation of physical dynamics for zero-shot generalization. However, they are computationally demanding and fundamentally constrained by the quality of generated videos, leading to poor out-of-domain generalization. Most related to our work, SuSIE (Black et al.) leverages 2D image-editing as visual guidance but relies on a goal-conditioned policy to implicitly interpret spatial relations, lacking explicit geometric grounding. Recent work GoalVLA (Chen et al., 2025a) similarly lifts generative subgoals into 3D transformations. However, GoalVLA decouples the scale factors of active and passive objects during 3D alignment, leading to inconsistent scene geometry and significant spatial offsets. In contrast, our unified scale constraint during 3D registration preserves global scene structure, enabling robust and accurate manipulation.

## 3 Method

LAMP leverages edited images as strong spatial priors, formulating manipulation as predicting inter-object 3D transformations (Sec. 3.1). As illustrated in Fig. 2, our framework comprises three stages: 1) **Perception** (Sec. 3.2) extracts 3D spatial priors to ground task intent; 2) **Reasoning** (Sec. 3.3) estimates the precise 3D transformation via noise-robust point cloud registration; and 3) **Execution** (Sec. 3.4) maps this transformation into a target pose for end-effector trajectory optimization.

### 3.1 Task Formulation from an Editing Perspective

We formulate robotic manipulation as predicting inter-object $SE(3)$ transformations via visual editing. Given an initial RGB-D observation $(I_{\text{obs}}, D)$ and a language instruction $\mathcal{L}$, our goal is to generate a 6-DoF end-effector trajectory $\tau$. We assume $\mathcal{L}$ specifies a subtask-level command, as long-horizon tasks can be decomposed by high-level planners (Song et al., 2023; Hu et al., a). Specifically, $\mathcal{L}$ defines the interaction between an active object $\mathcal{O}_a$ and a passive object $\mathcal{O}_p$, where $\mathcal{O}_p$ simply denotes the functionally coupled static environment (e.g., a drawer housing) in single-object scenarios. Leveraging the spatial priors from image editing, we cast the manipulation problem as estimating the target transformation $\mathbf{T}_a \in SE(3)$ that maps $\mathcal{O}_a$ from its observed state to the visually edited goal state.

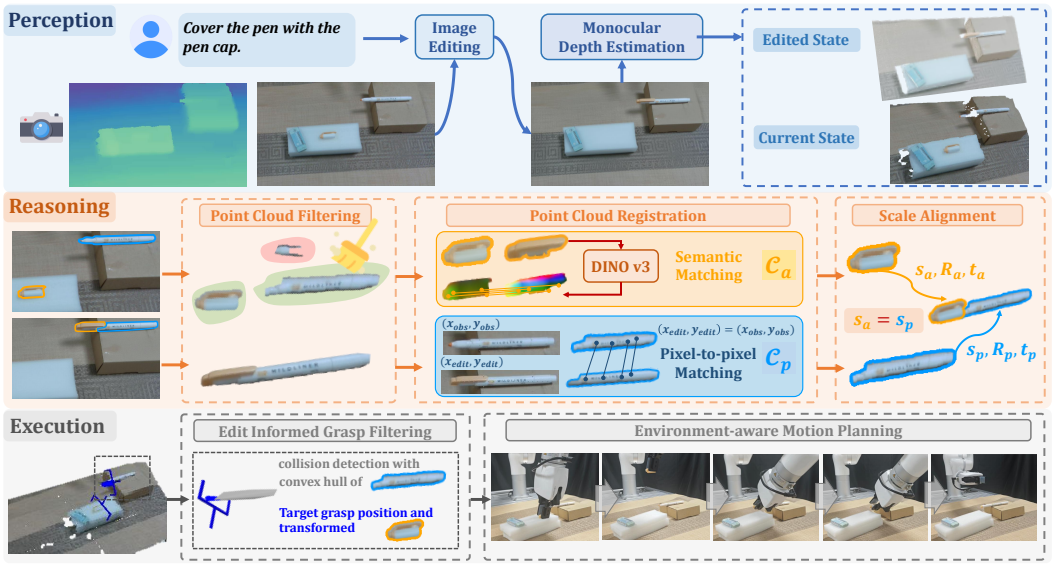



Figure 2: **Overview.** Given a single-view RGB-D observation and language instruction, our pipeline synthesizes a visual goal via image editing, robustly lifts it into an explicit inter-object $SE(3)$ transformation, and maps this continuous 3D prior into target poses for environment-aware execution.

### 3.2 Spatial Prior Extraction from Editing

Given an RGB observation $I_{\text{obs}} \in \mathbb{R}^{H\times W\times 3}$ and a task description $\mathcal{L}$, we generate an edited image $I_{\text{edit}}$ depicting the post-manipulation state of the active object $\mathcal{O}_a$ via image-editing models (Comanici et al., 2025; Wu et al., 2025; OpenAI, 2026). To recover its geometry, we lift $I_{\text{edit}}$ into a pixel-aligned point cloud $\mathcal{P}_{\text{edit}} \in \mathbb{R}^{(H\times W)\times 3}$ using a monocular depth estimator VGGT (Wang et al., 2025). To preserve spatial detail against resolution mismatch, we extract binary masks $\mathcal{M}^a_{\text{edit}}$ and $\mathcal{M}^p_{\text{edit}}$ of $\mathcal{O}_a$ and $\mathcal{O}_p$ from $I_{\text{edit}}$ via SAM3 (Carion et al., 2025), crop the enclosing bounding box, and resize it by $I_{\text{edit}}$ to match the estimator's input resolution before lifting to 3D.

### 3.3 Cross-State Point Cloud Registration

To estimate $\mathbf{T}_a$, we perform cross-state point cloud registration. While registration is well-established in 3D reconstruction (Zeng et al., 2017), aligning generative edited states introduces unique challenges: single-view observations are inherently noisy and incomplete (Fig. 3(b)), and interacting objects $(\mathcal{O}_a, \mathcal{O}_p)$ may move, deform or occlude each other (Fig. 3(a)). To overcome these issues, we propose a robust registration pipeline that sequentially filters unreliable geometries, performs object-centric alignment, and enforces a unified scale correction to preserve globally consistent spatial reasoning.

**Point Cloud Filtering.** RGB-D depth discontinuities inherently produce floating-edge artifacts (Fig. 3(b)) that severely degrade downstream registration. Standard density-based filters (e.g., DBSCAN (Schubert et al., 2017)) fail to remove them, as these artifacts remain locally dense and spatially proximal to valid regions. However, these points are highly separable in the semantic feature space (Fig. 3(c)). To exploit this, we propose a hierarchical 2D-3D filtering strategy: we first extract 2D features via DINOv3 (Siméoni et al., 2025) and group pixels by visual appearance using K-Means. We then apply DBSCAN within each semantic group (intra-cluster) and refine across them (inter-cluster). This fused filtering robustly isolates boundary artifacts, ensuring clean geometry for stable alignment (Fig. 3(e)).

**Point Cloud Registration.** We independently align the pixel-aligned observed ($\mathcal{P}^k_{\text{obs}}$) and edited ($\mathcal{P}^k_{\text{edit}}$) point clouds for both active and passive objects ($k \in \{a, p\}$), which are defined as:

$$\mathcal{P}^k_s = \{\, \mathbf{p}^s_i \in \mathbb{R}^3 \mid i \in \mathcal{M}^k_s \,\}, \quad s \in \{\text{obs}, \text{edit}\}. \tag{1}$$

The core challenge lies in establishing reliable cross-state correspondences $\mathcal{C}^k$. Since 2D image editing inherently preserves viewpoint and pixel-level consistency for static regions, the passive object $\mathcal{O}_p$ admits dense pixel-to-pixel matches:

$$\mathcal{C}^p = \left\{ (\mathbf{p}^{\text{obs}}_i, \mathbf{p}^{\text{edit}}_i) \mid i \in \mathcal{M}^p_{\text{obs}} \cap \mathcal{M}^p_{\text{edit}} \right\}. \tag{2}$$

Conversely, direct pixel alignment breaks down for the active object $\mathcal{O}_a$ due to editing-induced motions, deformations, or occlusions. Instead, we establish $\mathcal{C}^a$ via nearest-neighbor matching in the

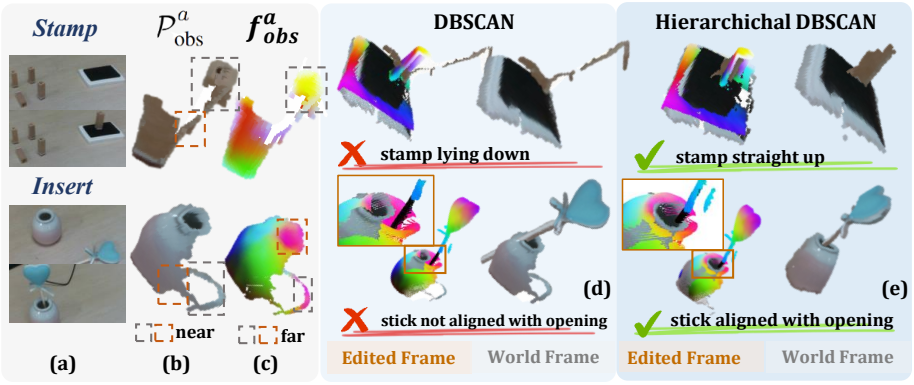



Figure 3: **2D-3D hierarchical point-cloud filtering.** (a) Observed (top) and edited (bottom) RGB inputs. (b) **Spatial space:** Flying-edge artifacts (gray) remain spatially proximal to valid points (orange). (c) **Feature space:** Artifacts are clearly separable from valid points when visualized via DINOv3 PCA (colorful points). (d) Standard spatial clustering fails to eliminate these locally dense artifacts, resulting in severe registration misalignment. (e) Our hierarchical filtering leverages 2D feature clusters to robustly isolate artifacts, recovering precise 3D spatial alignment.

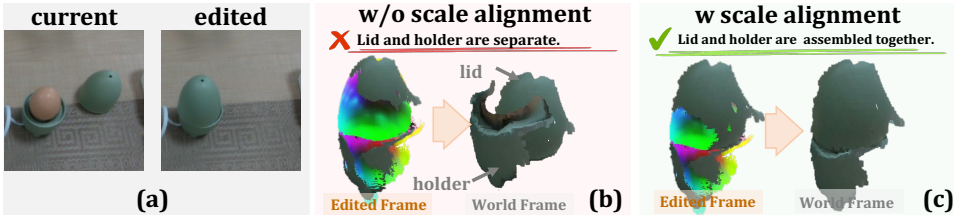



Figure 4: **Scale alignment.** **(a)** Inputs. **(b) w/o alignment:** Independent scale estimation causes severe spatial drift in the world frame. **(c) w/ alignment:** Enforcing a unified scale preserves structural consistency (Colorful points are $\mathcal{P}_{\text{edit}}$ with DINO features after PCA; green points are $\mathcal{P}_{\text{obs}}$).

DINOv3 (Siméoni et al., 2025) semantic feature space, which robustly encodes object-level identity. To reject outliers, we filter pairs exceeding a cosine distance threshold $d_{\text{thr}} = 0.3$:

$$\mathcal{C}^a = \left\{ (\mathbf{p}^{\text{obs}}_{i^*}, \mathbf{p}^{\text{edit}}_{j}) \;\middle|\; \begin{array}{l} j \in \mathcal{M}^a_{\text{edit}}, \\ i^* = \arg\min_{i \in \mathcal{M}^a_{\text{obs}}} d_{\cos}(f^{\text{obs}}_i, f^{\text{edit}}_j), \\ d_{\cos}(f^{\text{obs}}_{i^*}, f^{\text{edit}}_j) < d_{\text{thr}} \end{array} \right\}. \tag{3}$$

Given correspondences $\mathcal{C}^k$, we independently apply the Umeyama algorithm (Umeyama, 1991) to solve for the rotation $\mathbf{R}_k \in SO(3)$, translation $\mathbf{t}_k \in \mathbb{R}^3$, and scale $s_k \in \mathbb{R}_+$:

$$s_k \mathbf{R}_k \mathcal{P}^k_{\text{obs}} + \mathbf{t}_k \approx \mathcal{P}^k_{\text{edit}}. \tag{4}$$

Crucially, extracting explicit scale factors $s_k$ allows us to correct the inter-object scale inconsistencies inherently introduced by generative 2D image editing.

**Relative Transformation Computation.** Although registration yields transformations for both $\mathcal{O}_a$ and $\mathcal{O}_p$, their independent estimation inherently introduces scale inconsistencies ($s_a \neq s_p$). As illustrated in Fig. 4(b), this discrepancy causes spatial drift when projecting objects back to the world frame. Because pixel-to-pixel matching on $\mathcal{O}_p$ yields a highly reliable scale mapping, we treat it as the anchor ($s_a \leftarrow s_p$) and recompute $\mathbf{R}_a$ and $\mathbf{t}_a$ to enforce a unified global scale (Fig. 4(c)).

To obtain the final world-frame transformation $\mathbf{T}_a$ of the active object, we first extract its scale-free relative transformation $[\mathbf{R}^o_{a|p} \mid \mathbf{t}^o_{a|p}]$ with respect to $\mathcal{O}_p$ in the observation frame:

$$\mathbf{R}^o_{a|p} = \mathbf{R}_p^{-1}\mathbf{R}_a, \quad \mathbf{t}^o_{a|p} = \mathbf{R}_p^{-1}(\mathbf{t}_a/s_a - \mathbf{t}_p/s_p). \tag{5}$$

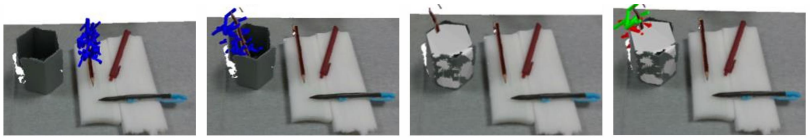
We then project the relative motion to the world frame using the camera extrinsics $[\mathbf{R}_{\text{o2w}} \mid \mathbf{t}_{\text{o2w}}]$:

$$\begin{aligned} \mathbf{R}^{\text{w}}_{a|p} &= \mathbf{R}_{\text{o2w}}\mathbf{R}^o_{a|p}\mathbf{R}^T_{\text{o2w}}, \\ \mathbf{t}^{\text{w}}_{a|p} &= \mathbf{R}_{\text{o2w}}\mathbf{t}^o_{a|p} + \mathbf{t}_{\text{o2w}} - \mathbf{R}^{\text{w}}_{a|p}\mathbf{t}_{\text{o2w}}. \end{aligned} \tag{6}$$

The final manipulation target is thus defined as $\mathbf{T}_a = [\mathbf{R}^{\mathrm{w}}_{a|p} \mid \mathbf{t}^{\mathrm{w}}_{a|p}]$.

### 3.4 Edited Goal Informed Execution

We decouple execution into grasping and trajectory planning. Off-the-shelf synthesizers like AnyGrasp (Fang et al., 2023) are task-agnostic, often yielding grasps that hinder subsequent interactions (e.g., grasping a pen by its tip prior to a tip-first insertion). To ensure task feasibility, we leverage the visual goal as a geometric prior: we project each candidate grasp into the target state via $\mathbf{T}_a$, and reject any pose that collides with the convex hull of the passive object $\mathcal{O}_p$ (Fig. 5). Finally, we deploy CuRobo (Sundaralingam et al., 2023) to generate collision-free robot trajectories within the voxelized environment.



**(a) grasp candidates** **(b) transformed** **(c) convex hull** **(d) filtered grasps**

Figure 5: **Edit-Informed Grasp Filtering.** **(a)** Task-agnostic candidate grasps (blue). **(b)** Candidates projected into the goal state via $\mathbf{T}_a$. **(c)** Collision convex hull (gray mesh) of $\mathcal{O}_p$. **(d)** Evaluated grasps: collisions are rejected (red), leaving valid task-specific grasps for execution (green).

## 4 Experiment

In this section, we evaluate LAMP and analyze its performance to address three key questions: **(1)** *High-Precision Alignment:* Can our continuous 3D representation achieve the fine-grained alignment required for complex object manipulation (Sec. 4.1)? **(2)** *Zero-Shot Generalization:* Does our object-centric spatial prior generalize better to novel tasks and unseen environments than robot-centric video baselines (Sec. 4.2)? **(3)** *Real-World Robustness:* How well does our method handle severe perceptual noise (e.g., depth artifacts) compared to sparse geometric proxies (e.g., 3D keypoints) (Sec. 4.3)? We validate LAMP in simulation (RoboTwin 2.0 (Chen et al., 2025b), CALVIN (Mees et al., 2022)) and real-world environments which use xArm7 and a single Intel RealSense D435i RGB-D camera.

### 4.1 Point Cloud Registration Accuracy

**Settings.** To evaluate spatial alignment accuracy, we curate a nine-task real-world suite captured via a single RGB-D camera; the complete task definitions are provided in Appendix Sec. A.10.

For the five-task quantitative subset in Tab. 1, we align scanned object meshes (Fig. 6) with pre-recorded human demonstrations using FoundationPose (Wen et al., 2024) to obtain reference 6-DoF transformations. Each quantitative task contains 10 distinct initial configurations.



Figure 6: **Scanned object meshes.**

**Baselines and Metrics.** We compare quantitatively with **2BY2** (Qi et al., 2025). **AnyPlace** (Zhao et al., 2025) is evaluated only qualitatively in Fig. 7, because its predictions on our partial OOD point clouds drift outside the physically meaningful range. We report translation RMSE in meters, **RMSE(t)**, and rotation RMSE in degrees, **RMSE(R)**. A dash indicates that the predicted transformation has an extremely large deviation and its error falls outside the meaningful reporting range.

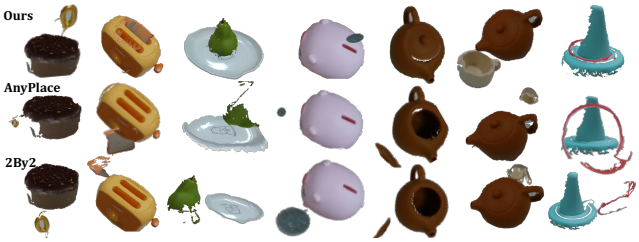



Figure 7: **Qualitative point-cloud alignment across seven tasks.** Rows show LAMP, AnyPlace, and 2BY2; columns show candle insertion, toast insertion, pear placing, coin insertion, lid covering, tea pouring, and ring stacking.

Table 1: **Spatial-alignment accuracy.** Translation RMSE $t$ (m) and rotation RMSE $R$ (deg) over 10 configurations per task. "–": prediction error is excessively large and outside the reporting range.

| Task | 2BY2 | | LAMP | |
|---|---|---|---|---|
| | $t$ | $R$ | $t$ | $R$ |
| Lid covering | 0.091 | 16.54 | **0.003** | **8.736** |
| Toast insertion | 0.095 | 35.12 | **0.015** | **11.10** |
| Block assembly | – | – | **0.005** | **30.05** |
| Tea pouring | – | – | **0.014** | **21.53** |
| Drawer opening | – | – | **0.017** | **2.614** |

**Results.** As shown in Tab. 1, LAMP achieves low translation and rotation errors on all five quantitative tasks. On lid covering and toast insertion, LAMP reduces both translation and rotation RMSE; on block assembly, tea pouring, and drawer opening, the baseline predictions deviate so severely that their errors fall outside the meaningful reporting range. The seven-task qualitative comparison in Fig. 7 further shows that LAMP preserves the instructed inter-object relation across insertion, placing, covering, pouring, and stacking tasks. In contrast, AnyPlace overfits to the clean point-cloud distribution seen during training: when applied to our noisy, single-view partial point clouds, its released checkpoint frequently predicts extreme, physically implausible transformations, revealing a substantial sim-to-real generalization gap.

### 4.2 Generalization in Simulation

**Settings.** To validate out-of-domain (OOD) generalization, we evaluate LAMP on two simulation benchmarks with 100 trials per task. We use the unseen CALVIN-D split (Mees et al., 2022) for block stacking, and RoboTwin 2.0 (Chen et al., 2025b) for single-arm and bimanual tasks under both *clean* and *randomized* scenarios.

**Metrics and Baselines.** We report all per-task success rates in Tabs. 2 and 3. We compare against three robot-centric generative baselines: **SuSIE** (Black et al.) (goal-conditioned policy), **VPP** (Hu et al., b; Shang et al., 2026) (video-latent policy), and **Vidar** (Feng et al., 2025a) (inverse dynamics from generated video). These baselines use task-specific training, whereas LAMP is evaluated zero-shot without task-specific robot data.

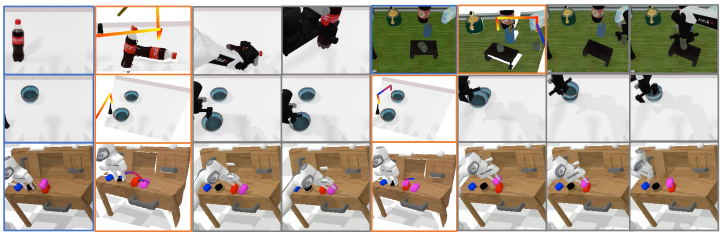


Figure 8: **Qualitative simulation rollouts.** Progression from edited goals (blue), via 3D spatial alignment and planned trajectories (orange), to execution (gray) across RoboTwin 2.0 (top two rows) and CALVIN-D (bottom row).

Table 2: **CALVIN-D success rates (%).** Results use 100 trials per task.

| Task | SuSIE | LAMP |
|---|---|---|
| Stack Two | 11 | **25** |
| Stack Three | 2 | **11** |
| Drawer Place | 58 | **71** |

**Results.** LAMP demonstrates superior zero-shot generalization. On RoboTwin 2.0, the video baselines degrade substantially in randomized scenes, whereas LAMP retains a 64.8% average success rate, compared with 13.3% for Vidar and 2.2% for VPP (Tab. 3). By distilling the task intent into an explicit, object-centric 3D transformation, our representation is more robust to background shifts and preserves fine-grained accuracy (e.g., *Adjust Bottle*); qualitative rollouts are provided in Fig. 8. Furthermore, on the CALVIN-D split, LAMP significantly outperforms SuSIE in strong spatial-reasoning tasks like *Stack Blocks*. Robot-centric policies overfit to 2D pixel coordinates and fail at 3D reasoning, whereas LAMP bypasses this by directly aligning 3D point clouds, proving profound zero-shot robustness of generative spatial priors. See Appendix Sec. A.15 and Figs. 15, 24 for image-editing failure and rejection analyses.

Table 3: **RoboTwin 2.0 success rates (%).** Results use 100 trials per task; "Rand." denotes randomized scenes.

| Task | VPP | | Vidar | | LAMP | |
|---|---|---|---|---|---|---|
| | Clean | Rand. | Clean | Rand. | Clean | Rand. |
| Adjust Bottle | 12 | 7 | 61 | 40 | **83** | **72** |
| Place Burger Fries | 3 | 0 | 93 | 15 | **98** | **97** |
| Place Object Stand | 5 | 1 | **90** | 10 | 89 | **79** |
| Stack Bowls Two | 0 | 0 | **91** | 14 | 83 | **71** |
| Stack Blocks Two | 3 | 0 | **58** | 16 | 54 | **37** |
| Stack Blocks Three | 0 | 0 | 10 | 3 | **41** | **33** |
| Blocks Ranking RGB | 0 | 0 | 50 | 4 | **56** | **50** |
| Move Pillbottle Pad | 0 | 0 | 71 | 4 | **89** | **82** |
| Move Playingcard Away | 11 | 6 | 96 | 22 | **98** | **97** |
| Place A2B Left | 10 | 4 | 85 | 8 | **87** | **43** |
| Place A2B Right | 11 | 7 | **91** | 17 | 89 | **42** |
| Place Bread Basket | 4 | 1 | **82** | 7 | 81 | **75** |
| **Average** | 4.9 | 2.2 | 73.2 | 13.3 | **79.0** | **64.8** |

### 4.3 Real-World Evaluation

**Settings.** We evaluate on 13 diverse real-world tasks (see the full list in Table 4) spanning *high-precision assembly* (insertion, covering, stacking), *articulated manipulation* (drawer/toaster opening), and *functional semantics* (pouring, cutting, placing), each with 10 randomized trials.

**Metrics and Baselines.** Tab. 4 shows task success rate. We compare LAMP against three state-of-the-art zero-shot baselines relying on discrete or sparse proxies: **VoxPoser** (Huang et al., 2023), which synthesizes trajectories via LLM-generated code and 3D value maps; **CoPA** (Huang et al., 2024), which uses VLMs to infer spatial constraints from interaction keypoints (provided with optimal masks for fairness); and **ReKep** (Huang et al., 2025), which optimizes trajectories using VLM-predicted relational keypoints as cost terms. We additionally compare against the concurrent transformation-based method **GoalVLA** (Chen et al., 2025a).

**Results.** LAMP exhibits superior task diversity and execution robustness. The failures of the baselines stem from three fundamental limitations: 1) **Inherently sparse language constraints**: They fail to express continuous geometric relations, causing systematic failures on high-precision tasks like insertion and covering (Fig. 9). 2) **Brittle sparse proxies**: ReKep is sensitive to sparse keypoint depth noise (Fig. 10). 3) **Inconsistent 3D scale**: GoalVLA suffers from spatial relationship drifts between interacting objects, strictly limiting fine-grained reliability. Conversely, LAMP lifts dense visual correspondences into full inter-object 3D transformations. By holistically aligning point clouds, our approach mitigates local depth errors and ensures robust geometric grounding.

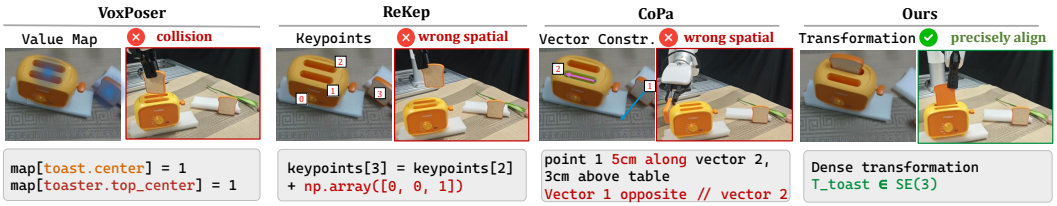



Figure 9: **Real-world *Toast Insertion*.** Sparse baselines fail to recover consistent geometric constraints, whereas LAMP estimates a precise 3D transformation.

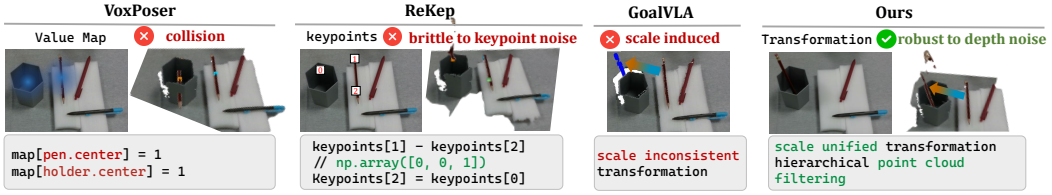



Figure 10: **Real-world *Pencil Insertion*.** ReKep is sensitive to depth noise and GoalVLA to scale drift, whereas LAMP remains accurate.

Table 4: Success rate of 13 real-world manipulation tasks. "/" indicates methods not applicable for the task.

| Tasks | VoxPoser | CoPA | ReKep | GoalVLA | Ours |
|---|---|---|---|---|---|
| Egg placing | 2/10 | 2/10 | 4/10 | 3/10 | **6/10** |
| Coin insertion | 0/10 | 0/10 | 0/10 | 2/10 | **5/10** |
| Pencil insertion | 0/10 | 4/10 | 3/10 | 1/10 | **7/10** |
| Toast insertion | 0/10 | 0/10 | 0/10 | 2/10 | **6/10** |
| Lid covering | 0/10 | 3/10 | 4/10 | 3/10 | **8/10** |
| Pen-cap covering | 0/10 | 1/10 | 2/10 | 0/10 | **6/10** |
| Tea pouring | 0/10 | 1/10 | 3/10 | 5/10 | **6/10** |
| Toast cutting | 0/10 | 0/10 | 5/10 | **8/10** | **8/10** |
| Block assembly | 0/10 | 1/10 | 0/10 | 0/10 | **6/10** |
| Ring stacking | 2/10 | 1/10 | 3/10 | 4/10 | **8/10** |
| Total | 4.0% | 13.0% | 24.0% | 28.0% | **66.0%** |
| Drawer opening | 2/10 | 4/10 | / | 3/10 | **6/10** |
| Drawer closing | 4/10 | 4/10 | / | 1/10 | **7/10** |
| Toaster opening | 2/10 | 1/10 | / | 4/10 | **5/10** |
| Total | 26.7% | 30.0% | / | 26.7% | **60.0%** |

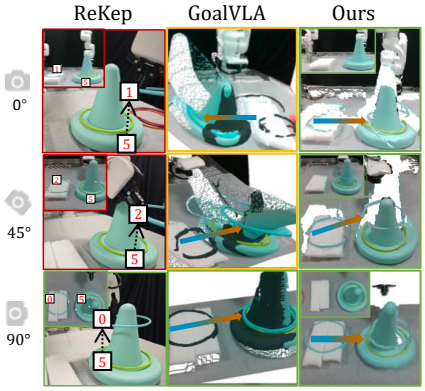



Figure 11: **Qualitative ring-stacking results under viewpoint changes.** Rows correspond to 0°, 45°, and 90° camera viewpoints; columns compare ReKep, GoalVLA, and LAMP.

### 4.4 Viewpoint Ablation

We evaluate the *ring-stacking* task from three camera viewpoints (0°, 45°, and 90°), using 10 trials per viewpoint. Table 5 quantitatively compares ReKep and CoPA and ablates LAMP's filtering and scale-alignment modules. Figure 11 provides the complementary qualitative comparison with ReKep and GoalVLA. Under severe viewpoint shifts, ReKep (Huang et al., 2025) misidentifies 2D keypoints, while GoalVLA's depth regression introduces visible 3D spatial drift. LAMP remains more stable, and the ablations show that filtering and scale alignment both contribute to this robustness.

Table 5: **Ring-stacking success rates under viewpoint changes** (10 trials per viewpoint). The quantitative baselines are ReKep and CoPA; the remaining columns ablate LAMP.

| Viewpoint | ReKep | CoPA | Ours (w/o filter) | Ours (w/o scale) | Ours (full) |
|---|---|---|---|---|---|
| 0° | 1/10 | 1/10 | 2/10 | 3/10 | **6/10** |
| 45° | 3/10 | 4/10 | 6/10 | 3/10 | **8/10** |
| 90° | 2/10 | 6/10 | 7/10 | 4/10 | **8/10** |

### 4.5 Ablation on Image-Editing Models

We compare Qwen-Image-Edit (Wu et al., 2025) and Gemini 2.5 Flash Image in Tab. 6. Gemini's stronger subject consistency and semantic adherence improve insertion and covering; Qwen performs better on tool use and ring stacking but is less reliable for directional relations and scene awareness. Since edits may retain the active object at its original location, we reject unreliable goals using the overlap between the edited mask and original object region. Appendix Fig. 17 provides qualitative comparisons.

Table 6: **Image-editor ablation** on five real-world tasks (10 trials per task).

| Task | Qwen | Gemini |
|---|---|---|
| Egg placing | **6/10** | 5/10 |
| Toast insertion | 1/10 | **6/10** |
| Pen-cap covering | 1/10 | **6/10** |
| Toast cutting | **8/10** | 5/10 |
| Ring stacking | **8/10** | 3/10 |

### 4.6 Long-horizon tasks and multimodal promptability

We design three long-horizon tasks: putting a duck into the red drawer, packing the eggs, and setting up the table. Fig. 12 shows example execution rollouts, highlighting that LAMP maintains accurate object alignment and successfully completes each subtask in sequence.

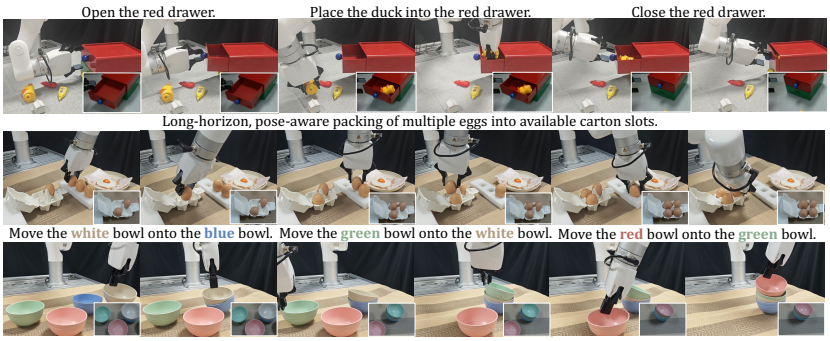



Figure 12: **Successful executions of the three manually decomposed multi-step tasks.** The inset in each sequence shows the edited subgoal state used for the corresponding execution stage.

## 5 Conclusion and Limitations

We present LAMP, a generalizable representation that lifts 2D image-editing priors into precise, inter-object 3D transformations for zero-shot robotic manipulation. By translating implicit spatial cues from generative models into continuous geometric constraints, LAMP inherently overcomes the brittleness of sparse symbolic proxies and the visual entanglement of 2D policies. We demonstrate robust, high-precision execution across diverse tasks, exhibiting resilience to viewpoint shifts and depth noise without requiring task-specific training data. Despite these promises, limitations remain. First, LAMP currently assumes rigid-body interactions, leaving the manipulation of deformable objects for future work. Second, ensuring inferred transformations kinematically reachable and collision-free introduces motion planning complexity. Third, as with most language-based models, it requires moderate prompt engineering when image editing exhibits hallucinations occasionally.

## AI USE STATEMENT

In this work, we used generative AI tools for language polishing, editing and formatting the manuscript source. All AI-assisted text and source edits were reviewed, revised, and verified by the authors. We take full responsibility for the final content of this work, including any text or artifacts produced with the aid of generative AI.

## REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando De Freitas. Playing hard exploration games by watching youtube. *Advances in neural information processing systems*, 31, 2018.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quan Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. In *Robotics: Science and Systems*, 2024.

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision (ECCV)*, 2024.

Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 1, pp. 348–353. IEEE, 2000.

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Rich Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pre-trained image-editing diffusion models. In *The Twelfth International Conference on Learning Representations*.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

Haonan Chen, Jingxiang Guo, Bangjun Wang, Tianrui Zhang, Xuchuan Huang, Boren Zheng, Yiwen Hou, Chenrui Tie, Jiajun Deng, and Lin Shao. Goal-vla: Image-generative vlms as object-centric world models empowering zero-shot robot manipulation. *arXiv preprint arXiv:2506.23919*, 2025a.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b.

Yun-Chun Chen, Haoda Li, Dylan Turpin, Alec Jacobson, and Animesh Garg. Neural shape mating: Self-supervised object assembly with adversarial shape priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12724–12733, 2022.

Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. Putting the object back into video object segmentation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3151–3161, 2024. doi: 10.1109/CVPR52733.2024.00304.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Karthik Dharmarajan, Wenlong Huang, Jiajun Wu, Li Fei-Fei, and Ruohan Zhang. Dream2flow: Bridging video generation and open-world manipulation with 3d object flow. *arXiv preprint arXiv:2512.24766*, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Ben Eisner and Harry Zhang. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *Robotics Science and Systems 2022*, 2022.

Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023.

Kuan Fang, Fangchen Liu, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *Robotics: Science and Systems (RSS)*, 2024.

Yao Feng, Hengkai Tan, Xinyi Mao, Guodong Liu, Shuhe Huang, Chendong Xiang, Hang Su, and Jun Zhu. Vidar: Embodied video diffusion model for generalist bimanual manipulation, 2025a. URL https://arxiv.org/abs/2507.12898.

Yao Feng, Chendong Xiang, Xinyi Mao, Hengkai Tan, Zuyue Zhang, Shuhe Huang, Kaiwen Zheng, Haitian Liu, Hang Su, and Jun Zhu. Vidarc: Embodied video diffusion model for closed-loop control. *arXiv preprint arXiv:2512.17661*, 2025b.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, 44(5): 701–739, 2025.

Chongkai Gao, Haozhuo Zhang, Zhixuan Xu, Cai Zhehao, and Lin Shao. Flip: Flow-centric generative planning as general-purpose manipulation world model. In *The Thirteenth International Conference on Learning Representations*.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, et al. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*, 2025.

Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5977–5984. IEEE, 2023.

Nicklas Hansen, Yixin Lin, Hao Su, Xiaolong Wang, Vikash Kumar, and Aravind Rajeswaran. Modem: Accelerating visual model-based reinforcement learning with demonstrations. In *The Eleventh International Conference on Learning Representations*.

Zhengdong Hong, Y Liu, H Hou, B Ai, J Wang, T Mu, Y Qin, J Gu, and H Su. Learning particle-based world model from human for robot dexterous manipulation. In *3rd RSS Workshop on Dexterous Manipulation: Learning and Control with Diverse Data*, 2025.

Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, a.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *Forty-second International Conference on Machine Learning*, b.

Haoxu Huang, Fanqi Lin, Yingdong Hu, Shengjie Wang, and Yang Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9488–9495. IEEE, 2024.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *Conference on Robot Learning*, pp. 4573–4602. PMLR, 2025.

Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. Strictly batch imitation learning by energy-based distribution matching. *Advances in Neural Information Processing Systems*, 33:7354–7365, 2020.

Kento Kawaharazuka, Tatsuya Matsushima, Andrew Gambardella, Jiaxian Guo, Chris Paxton, and Andy Zeng. Real-world robot applications of foundation models: A review. *Advanced Robotics*, 38(18):1232–1254, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning*, pp. 2679–2713. PMLR, 2025.

Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pp. 55–66. PMLR, 2023.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *CoRR*, 2024.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Thomas M Moerland, Joost Broekens, Aske Plaat, Catholijn M Jonker, et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023.

Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pp. 137–144, 2012.

Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: iterative visual prompting elicits actionable knowledge for vlms. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 37321–37341, 2024.

OpenAI. Gpt-image-2: Our most capable image generation model. 2026.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya 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)*, pp. 6892–6903. IEEE, 2024.

Mingjie Pan, Jiyao Zhang, Tianshu Wu, Yinghao Zhao, Wenlong Gao, and Hao Dong. Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17359–17369, 2025.

Shivansh Patel, Shraddhaa Mohan, Hanlin Mai, Unnat Jain, Svetlana Lazebnik, and Yunzhu Li. Robotic manipulation by imitating generated videos without physical demonstrations. In *Workshop on Foundation Models Meet Embodied Agents at CVPR 2025*.

Yu Qi, Yuanchen Ju, Tianming Wei, Chi Chu, Lawson LS Wong, and Huazhe Xu. Two by two: Learning multi-task pairwise objects assembly for generalizable robot manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17383–17393, 2025.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *Robotics: Science and Systems XIV*, 2018.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal conditioned imitation learning using score-based diffusion policies. In *Robotics: Science and Systems*, 2023.

Daniela Rus. In-hand dexterous manipulation of piecewise-smooth 3-d objects. *The International Journal of Robotics Research*, 18(4):355–381, 1999.

Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

Mingyo Seo, Steve Han, Kyutae Sim, Seung Hyeon Bang, Carlos Gonzalez, Luis Sentis, and Yuke Zhu. Deep imitation learning for humanoid loco-manipulation through human teleoperation. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2023.

Yu Shang, Zhuohang Li, Yiding Ma, Weikang Su, Xin Jin, Ziyou Wang, Lei Jin, Xin Zhang, Yinzhou Tang, Haisheng Su, et al. Worldarena: A unified benchmark for evaluating perception and functional utility of embodied world models. *arXiv preprint arXiv:2602.08971*, 2026.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6394–6400. IEEE, 2022.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2998–3009, 2023.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. In *7th Annual Conference on Robot Learning*.

Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8112–8119. IEEE, 2023.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024a.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024b.

Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *Conference on Robot Learning*, pp. 306–316. PMLR, 2018.

S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. doi: 10.1109/34.88573.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Song XiXuan, et al. Cogvlm: Visual expert for pretrained language models. *Advances in Neural Information Processing Systems*, 37:121475–121499, 2024.

Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17868–17879, 2024.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Ruihai Wu, Chenrui Tie, Yushi Du, Yan Zhao, and Hao Dong. Leveraging se (3) equivariance for learning 3d geometric shape assembly. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14311–14320, 2023.

Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Iurii Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: 3d point tracking made easy. In *ICCV 2025 Workshop on Wild 3D: 3D Modeling, Reconstruction, and Generation in the Wild*.

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20406–20417, 2024.

Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4737–4746, 2023.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pp. 537–546. PMLR, 2022.

Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1802–1811, 2017.

Yuchi Zhao, Miroslav Bogdanovic, Chengyuan Luo, Steven Tohme, Kourosh Darvish, Alan Aspuru-Guzik, Florian Shkurti, and Animesh Garg. Anyplace: Learning generalizable object placement for robot manipulation. In *Conference on Robot Learning*, pp. 4038–4057. PMLR, 2025.

Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. Tesseract: learning 4d embodied world models. *arXiv preprint arXiv:2504.20995*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

# A Appendix

## A.1 Closed-loop Manipulation

To further demonstrate how our extracted 3D priors support closed-loop manipulation, we evaluate LAMP on the *Lid covering* task under human-induced disturbances, where the passive object is moved during execution (Fig. 13). We use Cutie (Cheng et al., 2024) to track the mask of the active object $\mathcal{O}_a$ across frames. A straightforward approach is to track keypoints (Xiao et al., 2024; Xiao et al.) inside the mask to obtain point-to-point correspondences, but current keypoint trackers are insufficiently accurate, particularly under rotation, resulting in unreliable 3D alignment. In contrast, dense pixel-wise matching with DINO features provides robust correspondences, enabling a more precise estimation of the active object's transformation for closed-loop control.

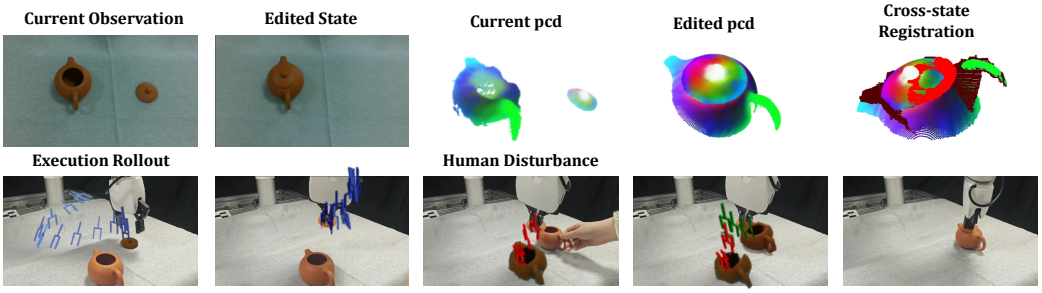



Figure 13: **Visualization of closed-loop execution rollout.**

## A.2 Runtime Profiling

To analyze the computational overhead of our multi-module pipeline, we conducted runtime profiling as illustrated in Fig. 14. Adhering to a 'think-before-act' paradigm, computationally intensive modules are executed outside the primary control loop. Consequently, while perception remains efficient, the overall latency is primarily dominated by the image editing querying phase.

## A.3 System Error Breakdown

We conduct an empirical investigation of system errors by analyzing the failure cases from Tab. 4. As illustrated in Fig. 15, the majority of failures are attributed to the image editing module. These cases typically involve unintended modifications to task-irrelevant scene elements or a failure to reflect the

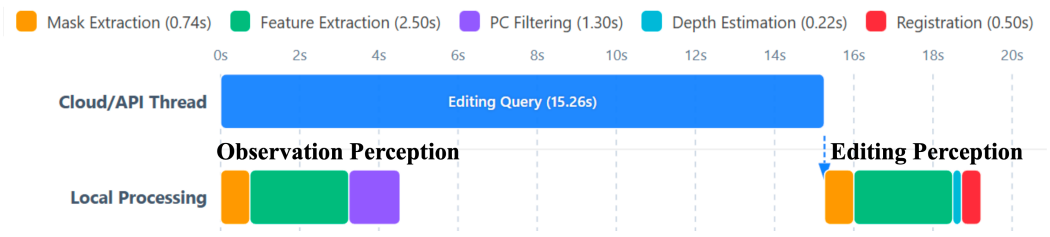



Figure 14: **Runtime profiling.**

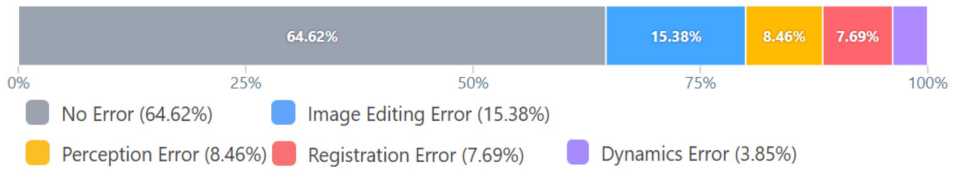



Figure 15: **System error breakdown.**

requested edits in the output. Perception and registration errors constitute another significant portion. These failures are predominantly triggered by small-scale objects or severe viewpoint occlusions, both of which hinder accurate spatial reasoning. In contrast, the low-level controller contributes only a minimal fraction, indicating that once a valid plan is generated, the execution remains relatively robust.

### A.4 Pseudo-code for Hierarchical Point Cloud Filtering

As shown in Algo. 1, given the object point cloud $\mathcal{P}_{\text{obs}}^{a/p}$ projected from the current RGB-D observation, the corresponding DINO feature $\mathbf{F}_{\text{obs}}$ and the object mask $\mathcal{M}_{\text{obs}}^{a/p}$, the algorithm outputs a filtered set of valid 3D points along with a pixel-aligned binary mask indicating the retained regions.

### A.5 Implementation Details for Spatial Prior Extraction from Editing

Resolution mismatch between $I_{\text{edit}}$ and the depth estimator may cause spatial detail loss if directly processed. To mitigate this, we extract binary masks $\mathcal{M}_{\text{edit}}^{a}$ and $\mathcal{M}_{\text{edit}}^{p}$ of $\mathcal{O}_a$ and $\mathcal{O}_p$ from $I_{\text{edit}}$, using SAM3 (Carion et al., 2025). For single-object instructions (e.g., "open the red drawer"), the passive object $\mathcal{O}_p$ denotes its functionally coupled static surroundings (e.g., the drawer housing). We then crop the tight bounding box enclosing $\mathcal{M}_{\text{edit}}^{a}$ and $\mathcal{M}_{\text{edit}}^{p}$, and resize or pad it by the original $I_{\text{edit}}$ to match the estimator's input resolution. For resized images, the predicted depth is upsampled back to the cropped RGB resolution, ensuring one-to-one pixel correspondence. This preserves spatial detail and yields accurate 3D grounding of manipulated regions in $\mathcal{M}_{\text{edit}}^{a} \cup \mathcal{M}_{\text{edit}}^{p}$ in $I_{\text{edit}}$.

### A.6 Implementation Details for Point Cloud Registration

Since the DINO feature-based matching for the active object requires KNN to compute the distance matrix, we use the `cuml` library to accelerate the computation.

### A.7 More Results for Comparison

While the concurrent work GoalVLA (Chen et al., 2025a) adopts a pipeline similar to ours, it overlooks the critical challenges of depth alignment and point cloud registration essential for fine-grained manipulation. This oversight leads to significantly lower success rates in precision-demanding tasks such as assembly and insertion, as quantified in Tab. 4.

We emphasize that achieving precise scale alignment between the edited and observed images is the key to lifting 2D edits into a reliable 3D prior for manipulation. In our registration process, we enforce the constraint $s_a = s_p$ to ensure that when the objects are transformed back to world

**Algorithm 1:** Hierarchical Point Cloud Filtering

**Input:** object point cloud $\mathcal{P}_{\text{obs}}^{a/p} \in \mathbb{R}^{M\times 3}$, DINO features of the point cloud $\mathbf{F}_{\text{obs}}^{a/p} \in \mathbb{R}^{M\times D}$, Number of K-Means layers $K$, DBSCAN params $(\varepsilon, \text{MinPts})$

**Output:** filtered point cloud $\mathcal{P'}_{\text{obs}}^{a/p} \in \mathbb{R}^{N\times 3}$, corresponding mask $\mathcal{M'}_{\text{obs}}^{a/p} \in \mathbb{B}^{M}$ of chosen area

▷ Initialization

$\mathbf{L} \leftarrow -1 \in \mathbb{Z}^{M\times 1}$; gid $\leftarrow 0$

▷ Stage 1: Feature Scaling

$\mathbf{F} \leftarrow \text{Standardize}(\tilde{\mathbf{F}})$;

▷ Stage 2: Intra-cluster Filtering

$\mathbf{L}_{\tilde{\mathbf{F}}} \leftarrow \text{KMeans}(\tilde{\mathbf{F}}, K)$; ▷ Feature Layering

**for** $k = 0$ **to** $K-1$ **do**

  $\mathcal{I}_k \leftarrow \{i \mid \mathbf{L}_{\tilde{\mathbf{F}}}[i] = k\}$

  **if** $|\mathcal{I}_k| <$ *MinPts* **then**

    **continue**

  **end**

  $\mathcal{P}_k^{a/p} = \mathcal{P}_{\text{obs}}^{a/p}[\mathcal{I}_k]$

  $\mathbf{Y}_k \leftarrow \text{DBSCAN}(\mathcal{P}_k; \varepsilon, \text{MinPts})$

  Let $s_c$ be the size of cluster $c \neq -1$

  **if** *no valid cluster* **then**

    **continue**

  **end**

  $c^\star \leftarrow \arg\max_c s_c$ ▷ dominant DBSCAN cluster

  **if** $s_{c^\star} \geq S_{\min}$ **then**

    Assign global cluster:

      $\mathcal{J} = \{i \in \mathcal{I}_k \mid \mathbf{Y}_k[i] = c^\star\}$

      $\mathbf{L}[i] \leftarrow \text{gid} \quad \forall i \in \mathcal{J}$

    gid $\leftarrow$ gid $+ 1$

  **end**

**end**

$\mathcal{M}_{\text{intra}}^{a/p} \leftarrow \mathbf{L} \neq -1$

▷ Stage 3: Inter-cluster Filtering

$\mathcal{P}_{\text{intra}}^{a/p} \leftarrow \mathcal{P}_{\text{obs}}^{a/p}[\mathcal{M}_{\text{intra}}^{a/p}]$

$\mathbf{Y}_{\text{intra}} \leftarrow \text{DBSCAN}(\mathcal{P}_{\text{intra}}^{a/p})$;

Let $s_c$ be cluster sizes for all $c \neq -1$

$c^\star \leftarrow \arg\max_c s_c$

$\mathcal{M}_{\text{inter}}^{a/p} \leftarrow \mathbf{Y}_{\text{inter}} = c^\star$

$\mathcal{M'}_{\text{obs}}^{a/p} \leftarrow \mathcal{M}_{\text{intra}}^{a/p}; \mathcal{M'}_{\text{obs}}^{a/p}[\mathcal{M}_{\text{inter}}^{a/p}] = \mathcal{M}_{\text{inter}}^{a/p}$

**return** $\mathcal{P'}_{\text{obs}}^{a/p} \leftarrow \mathcal{P}_{\text{intra}}^{a/p}[\mathcal{M}_{\text{inter}}^{a/p}]$ and $\mathcal{M'}_{\text{obs}}^{a/p}$

coordinates, the spatial relationship between the active and passive objects is strictly preserved.

$$s_{a/p} \cdot \mathbf{R}_{a/p} \mathcal{P}_{\text{obs}}^{a/p} + \mathbf{t}_{a/p} \approx \mathcal{P}_{\text{edit}}^{a/p}. \tag{7}$$

In contrast, GoalVLA (Chen et al., 2025a) aligns edited images with observations via depth linear regression, computing the transformation of the active object under a optimized scale $s$. Since physical objects are non-deformable, ignoring the consistency of $s$ and relying solely on $R$ and $T$ causes a shift in the relative spatial configuration. While such offsets may be negligible for coarse tasks like pick-and-place as in their evaluations, even a 1% scale error can result in significant translation offsets that are catastrophic for fine-grained manipulation as shown in Fig. 16.

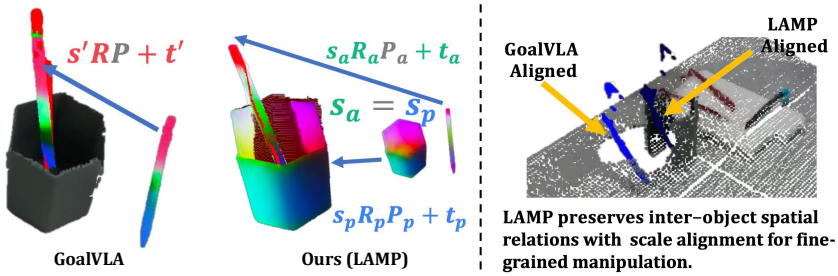



Figure 16: **Comparison of Alignment with GoalVLA.** While GoalVLA (Chen et al., 2025a) treats the passive object (e.g., the gray holder) as a static reference and aligns the active object (pencil) using an independent scale $s'$, this decoupling fails to account for global scene consistency. As illustrated in the right figure, such independent scaling distorts the relative spatial relationship between the two objects when transformed back to world coordinates. In contrast, our method enforces a unified scale across both the pencil and holder, strictly preserving their spatial configuration and ensuring the pencil remains correctly centered within the holder in 3D space.

### A.8 Qualitative Comparison of Editing Models

Figure 17 provides additional qualitative comparisons between Qwen-Image-Edit and Gemini 2.5 Flash; the corresponding quantitative ablation is reported in Sec. 4.5 and Tab. 6.

### A.9 More Visualization Results

More visualization results for cross-state point cloud registration and edited-informed grasping are shown in Fig. 19.

### A.10 Task Details for Spatial Alignment

We collect a nine-task real-world RGB-D suite for methods that take two object point clouds and predict an inter-object transformation. Figure 7 visualizes seven tasks, whereas Tab. 1 reports the complete five-task quantitative subset for which repeatable reference transformations are available. The union of these subsets comprises the nine tasks described below.

**Candle insertion**: $\mathcal{O}_a$ is the candle and $\mathcal{O}_p$ is the cake. $\mathcal{L}$ refers to "insert the candle onto the cake". The goal is to insert the candle anywhere on the cake surface.

**Toast insertion**: $\mathcal{O}_a$ is the toast and $\mathcal{O}_p$ is the toaster. $\mathcal{L}$ refers to "insert the toast into the toaster". The goal is to insert the toast into any valid slot of the toaster.

**Coin insertion**: $\mathcal{O}_a$ is the coin and $\mathcal{O}_p$ is the piggy bank. $\mathcal{L}$ refers to "insert the coin into the piggy bank". The goal is to align the coin with the bank's slot and orient it correctly for insertion.

**Pear placing**: $\mathcal{O}_a$ is the pear and $\mathcal{O}_p$ is the plate. $\mathcal{L}$ refers to "place the pear on the plate" . The goal is to place the pear anywhere on the plate in any stable orientation.

**Lid covering**: $\mathcal{O}_a$ is the lid and $\mathcal{O}_p$ is the teapot. $\mathcal{L}$ refers to "cover the teapot with the lid". The goal is to place the lid onto the teapot opening with proper alignment.

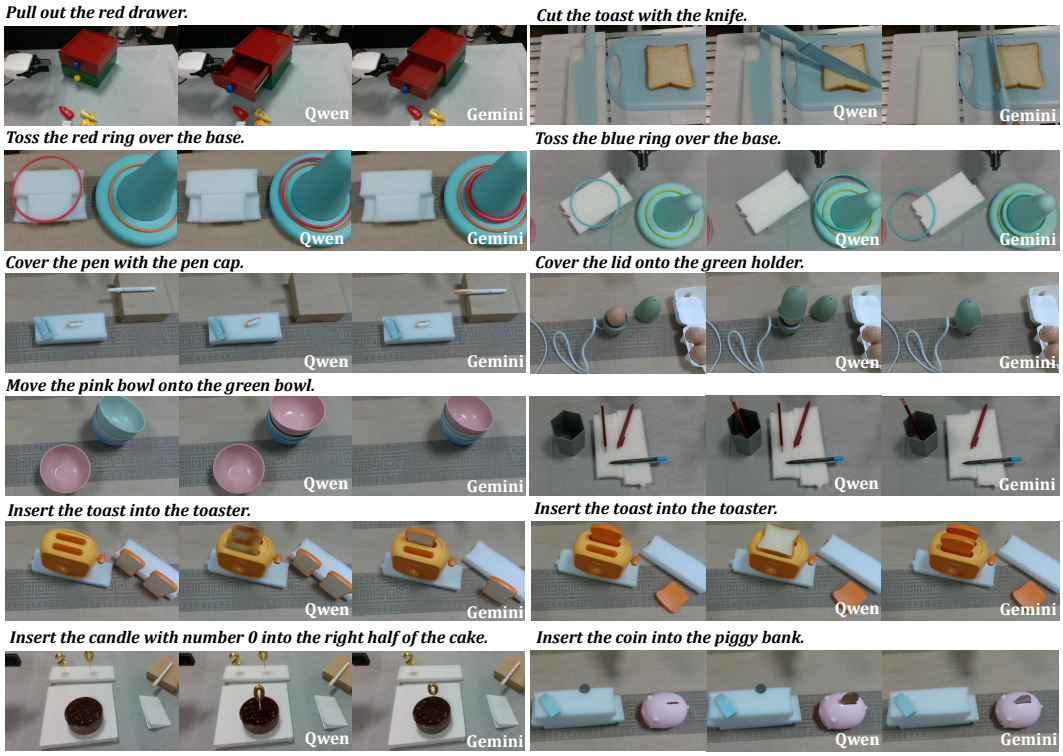



Figure 17: **Comparison of edited manipulation state with different editing models.**

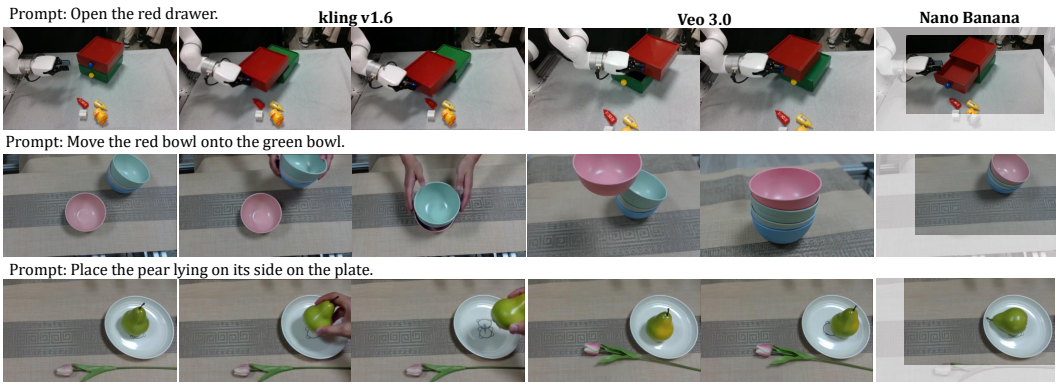



Figure 18: **Comparison between edited-image priors and video-generation priors for long-horizon manipulation.** Edited-image priors provide stronger semantic adherence with better subject and background consistency.

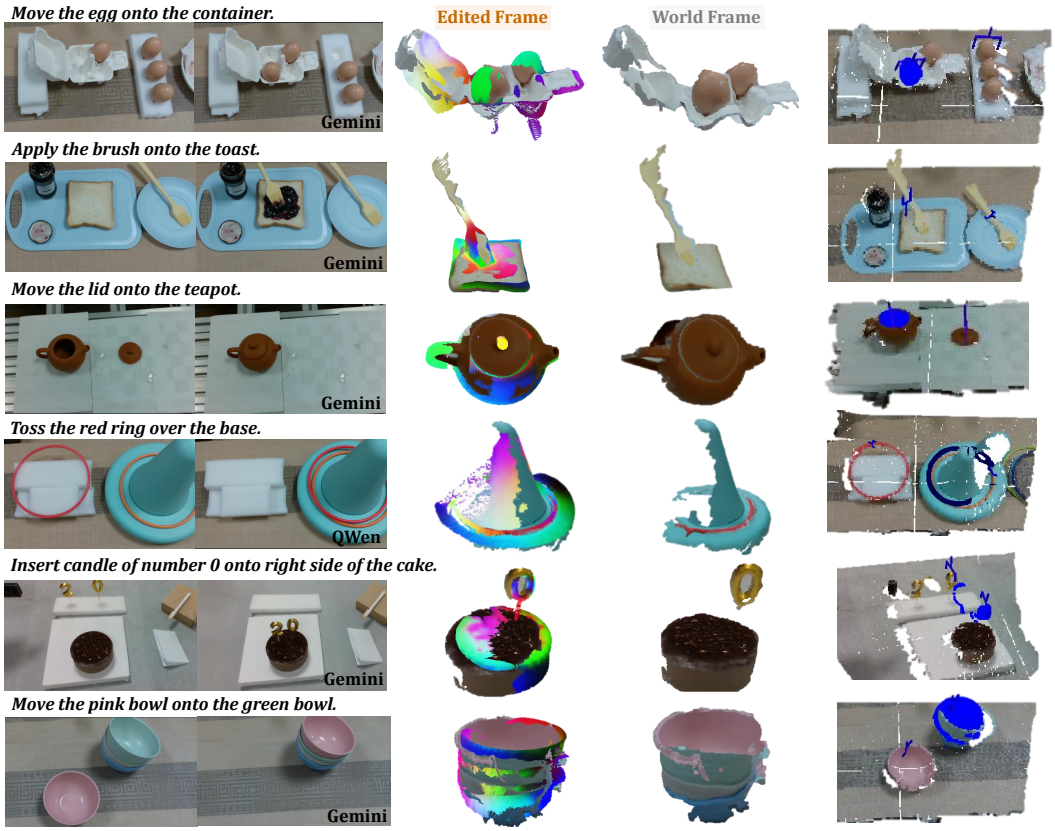



Figure 19: **More visualization results.** The first and second columns show the original observation and the edited state. The third and fourth columns show the registered point clouds in the edited frame and the world frame. The colored point clouds are $\mathcal{P}_{\text{edit}}^{a/p}$ after PCA. The last column shows the filtered grasp and the transformed active object.

**Tea pouring**: $\mathcal{O}_a$ is the teapot and $\mathcal{O}_p$ is the cup. $\mathcal{L}$ refers to "pour tea from the teapot into the cup". The goal is to rotate and position the teapot such that the spout aligns with and tilts over the cup.

**Ring stacking**: $\mathcal{O}_a$ is the ring and $\mathcal{O}_p$ is the base. $\mathcal{L}$ refers to "stack the ring onto the base". The goal is to align the ring hole with the peak of the base and then move the ring down to put them in place.

**Block assembly**: $\mathcal{O}_a$ is a block and $\mathcal{O}_p$ is another block or base structure. $\mathcal{L}$ refers to "assemble the two blocks together". The goal is to align their contact surfaces.

**Drawer opening**: $\mathcal{O}_a$ is the drawer and handle and $\mathcal{O}_p$ is the fixed drawer frame. $\mathcal{L}$ refers to "open the drawer". The goal is to translate the drawer outward along its rail direction.

For a fair comparison, we train **2BY2** using its official configuration and recenter all input point clouds at the origin for inference. We evaluate **AnyPlace** using its publicly released checkpoint and include its outputs in the qualitative comparison in Fig. 7. We observe that AnyPlace is strongly overfit to its clean training point-cloud distribution: on noisy, single-view partial point clouds, the checkpoint often outputs extreme translations or rotations that move the active object far outside a physically plausible placement region. This distribution mismatch makes its numerical errors uninformative and prevents a shared task-conditioned quantitative evaluation; therefore, the quantitative table compares LAMP with 2BY2 on the subset with a common evaluation interface.

### A.11 Details for Open-world Manipulation

For each task, we rearrange the objects across 10 trials and ensure that they remain within the robot's reachable and kinematically feasible workspace. To maintain identical initial configurations across baselines, we manually reset the scene after each execution. Success rates are evaluated according to the task-specific criteria described below.

**Egg placing**: The environment includes an egg ($\mathcal{O}_a$) and an egg holder ($\mathcal{O}_p$), with task description $L$ "move the egg onto the green egg holder". The task involves grasping the egg, aligning it with the holder and placing it stably onto the holder. The success criterion requires the egg resting upright on the holder without rolling or flipping.

**Coin insertion**: The environment includes a coin ($\mathcal{O}_a$) and a piggy bank ($\mathcal{O}_p$), with task description $L$ "insert the coin into the piggy bank". The task includes grasping the coin, aligning it with the slot of the piggy bank and inserting it into the slot. The success criterion requires successfully inserting the coin into the piggy bank through the slot.

**Pencil insertion**: The environment includes a pencil ($\mathcal{O}_a$) and a pencil holder ($\mathcal{O}_p$), with task description $L$ "insert the pencil into the holder". The task includes grasping the pencil, aligning it with the opening of the holder, and inserting it vertically into the holder. The success criterion requires the pencil standing stably inside the holder.

**Toast insertion**: The environment includes a toast ($\mathcal{O}_a$) and a toaster ($\mathcal{O}_p$), with task description $L$ "insert the toast into the toaster". The task includes grasping the toast, aligning it with a toaster slot, and inserting it. The success criterion requires the toast fully slid into a slot of the the toaster.

**Lid covering**: The environment includes a lid ($\mathcal{O}_a$) and a teapot ($\mathcal{O}_p$), with task description $L$ "cover the teapot with the lid". The task includes grasping the lid, aligning it with the teapot opening, and placing it. The success criterion requires the lid fitting the teapot perfectly.

**Pen-cap covering**: The environment includes a pen cap ($\mathcal{O}_a$) and a pen body ($\mathcal{O}_p$), with task description $L$ "cover the pen with the pen cap". The task includes grasping the pen cap, aligning it with the pen tip of the pen body, and cover the pen cap onto the pen tip. The success criterion requires the the pen cap fully attaching to the pen.

**Tea pouring**: The environment includes a teapot ($\mathcal{O}_a$) and a cup ($\mathcal{O}_p$), with task description $L$ "pour the tea from the teapot into the cup". The task includes grasping the teapot, tilt it over the cup, and maintaining the control. The success criterion requires the the water visibly flowing into the teacup from the teapot.

**Toast cutting**: The environment includes a knife ($\mathcal{O}_a$) and a toast ($\mathcal{O}_p$), with task description $L$ "cut the toast with the knife". The task includes grasping the knife, aligning it with the toast, and cutting along a straight trajectory. The success criterion requires a visible cut edge made through the toast.

**Block assembly**: The environment includes a block placed at the right hand ($\mathcal{O}_a$) and another matched block placed at the left hand($\mathcal{O}_p$), with task description $L$ "assemble the right block to the left block". The task includes grasping the right block, aligning it with the left block, and assembling it. The success criterion requires a the block fitted correctly with another block.

**Ring stacking**: The environment includes a ring ($\mathcal{O}_a$) and base with peak ($\mathcal{O}_p$), with task description $L$ "insert the ring onto the base". The task includes grasping the ring, aligning it with the peak of the base, and lowering it. The success criterion requires the ring fully placed onto the peak of the base.

**Drawer opening**: The environment includes a red drawer with handle ($\mathcal{O}_a$) (the drawer frame as $\mathcal{O}_p$), with task description $L$ "open the red drawer". The task includes grasping the handle and pulling the drawer outward along its rail direction. The success criterion requires the drawer opens beyond a predefined threshold (i.e., 10cm).

**Drawer closing**: The environment includes the same drawer as **Drawer opening** but initially open, with task description $L$ "close the red drawer". The task includes pushing the opened drawer along its rail. The success criterion requires the drawer opens pushed within a predefined threshold (i.e., 2cm).

**Toaster opening**: The environment includes the slide button of a toaster ($\mathcal{O}_a$) (the rest part of the toaster as $\mathcal{O}_p$), with task description $L$ "slide the button of the toaster downwards". The task includes sliding down the button of the toaster along its rail. The success criterion requires the button of the toaster slid steadily and completely down.

To ensure a fair comparison with the baselines, we focus primarily on the interaction between the objects. Given the same RGB-D observations, we use GPT-4o to extract manipulation constraints for **VoxPoser**, **ReKep**, and **CoPA**. For all baselines, we provide the best available object masks and identical task instructions. Since accurate keypoint localization in the real world depends heavily on point-cloud quality, the qualitative comparisons in Figs. 9 and 10 provide each baseline with the correct keypoint locations to better isolate the robustness of its extracted constraints.

### A.12 Details for Long-horizon Manipulation

For the long-horizon manipulation, we design three tasks as detailed below.

**Putting a duck into the red drawer**: The environment contains stacked drawers (the red drawer on top of the green one), along with a duck and other toys on the table. The task consists of three stages: (i) opening the red drawer, (ii) placing the duck inside, and (iii) closing the red drawer.

**Packing up the eggs**: The environment contains three eggs standing upright in a row and an egg container with one egg already packed. The task consists of three stages, each involving picking up an egg from the row and placing it into an available slot in the container.

**Setting up the table**: The environment contains four bowls on the table colored with white, blue, red and green. The task requires stacking them sequentially according to the color order specified in the instruction.

### A.13 Simulation Experiments

**Task Descriptions.** We evaluate LAMP on a set of RoboTwin 2.0 (Chen et al., 2025b) simulation tasks that require spatial rearrangement, stacking, insertion-like placement, and object reorientation. The tasks are summarized as follows.

- *Adjust Bottle*: pick up a bottle lying on the table with the appropriate arm and stand it upright at a target position while preserving the bottle's head-up orientation.
- *Blocks Ranking RGB*: arrange the red, green, and blue blocks into a row from left to right in the order red–green–blue.

- *Move Pillbottle Pad*: pick up the pill bottle with the appropriate arm and place it onto a blue pad while keeping the pad stationary.
- *Move Playingcards Away*: pick up the playing cards and move them farther toward the same table edge on which they initially appear.
- *Place A2B Left*: pick up object A and place it to the left of object B while leaving object B at its original location.
- *Place A2B Right*: pick up object A and place it to the right of object B while leaving object B at its original location.
- *Place Bread Basket*: pick up one or two bread pieces from the table and place them inside the bread basket; when two breads are present, both pieces must be transferred into the basket.
- *Place Burger Fries*: use the two arms to transfer a hamburger and a box of french fries onto a tray, placing the hamburger on the left side of the tray and the fries on the right side.
- *Place Object Stand*: pick up a randomly selected tabletop object with the appropriate arm and place it onto a display stand while keeping the stand fixed.
- *Stack Bowls Two*: pick up one of two bowls and place it on top of the other bowl, using the relative initial layout to decide which bowl should be moved.
- *Stack Blocks Two*: move a red block to the bottom-center target location and then place a green block on top of it, forming a two-block stack.
- *Stack Blocks Three*: sequentially move and stack three colored blocks to form a stable three-block tower.

We additionally evaluate LAMP on CALVIN (Mees et al., 2022) simulation tasks with long-horizon object manipulation. The CALVIN tasks are summarized as follows.

- *Stack blocks two*: pick up one block from the table and place it on top of another block to form a stable two-block stack.
- *Stack blocks three*: sequentially pick and place blocks to build a stable three-block stack.
- *Open drawer, place in drawer, and close drawer*: open the drawer, grasp the target block, place it into the drawer, and then close the drawer.

The complete RoboTwin 2.0 and CALVIN-D quantitative results have been moved to Tabs. 3 and 2 in the main paper. We provide task definitions and additional rollout visualizations here.

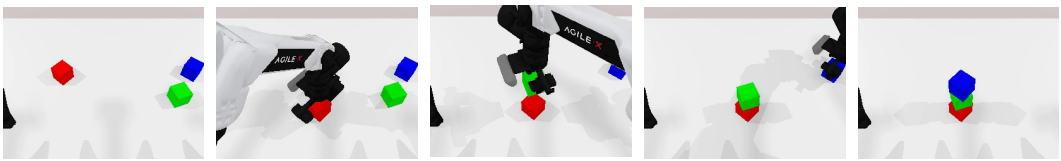



Figure 20: **Successful RoboTwin 2.0 rollout on *Stack Blocks Three*.** Keyframes from one execution show LAMP sequentially relocating the blocks and completing a stable three-block tower.

### A.14 More Comparisons with Baselines

**Visual Action Baselines.** We compare against robot-centric generative action baselines that use visual generative models as policy priors. VPP (Hu et al., b) follows the video-latent policy paradigm. In our RoboTwin 2.0 experiments, its video prior is based on Wan2.2 TI2V-5B (Wan et al., 2025) fine-tuned on the RoboTwin 2.0 dataset. The action DiT is then task-specifically fine-tuned on the evaluated RoboTwin tasks for 200 epochs before testing. Given the current observation and task instruction, VPP predicts one-step forward visual dynamics and conditions the policy on the resulting latent representation to produce robot actions. This allows VPP to exploit temporal priors from large-scale video models, but the representation remains primarily robot- and image-centric. When the scene appearance is randomized or the generated future frames contain visual artifacts, the extracted policy features may no longer correspond to metric object geometry, leading to brittle execution.

Vidar (Feng et al., 2025a) uses generated videos together with masked inverse dynamics. It first synthesizes task-relevant future visual states, then infers robot actions from the generated visual trajectory, focusing on manipulated regions through masks. This improves over directly conditioning

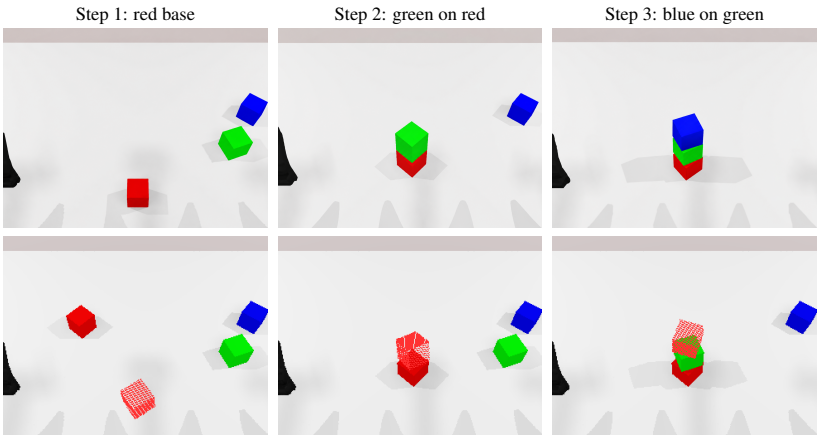



Figure 21: **Pipeline visualization for the *Stack Blocks Three* rollout.** Top row: edited subgoal images for the three sequential manipulation steps. Bottom row: the corresponding perception and registration outputs, where the transformed active-object point cloud is overlaid on the current observation to recover the next 3D target.




Figure 22: **Successful RoboTwin 2.0 rollout on *Place Shoe*.** Keyframes show LAMP grasping the shoe, moving it above the blue target block, and placing it horizontally on the target.

on raw generated frames, but the final control signal is still derived from 2D temporal visual consistency. As a result, Vidar can perform well in clean scenes where generated videos are visually consistent, but it is sensitive to background changes, object appearance shifts, camera viewpoints change and hallucinated intermediate states.

SuSIE (Black et al.) is the closest baseline to our use of image editing. It also generates edited subgoal images from language instructions, but uses them as visual goals for a learned goal-conditioned policy. Thus, SuSIE relies on the policy to implicitly infer the required spatial relation from pixels and convert it into actions. In contrast, LAMP explicitly lifts the edited visual prior into an inter-object 3D transformation before planning, making the manipulation target metric, object-centric, and less dependent on the policy's training distribution.

**VLM Baselines.** VoxPoser (Huang et al., 2023) uses LLM-generated code to compose 3D value maps from language, visual detections, and geometric heuristics. The robot trajectory is optimized

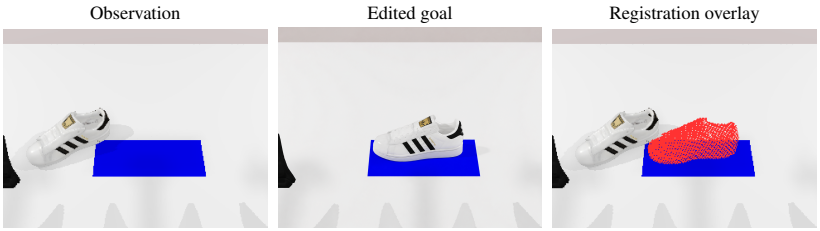



Figure 23: **Pipeline visualization for the *Place Shoe* rollout.** The edited image specifies the target shoe pose on the blue block, and the perception/registration output recovers the corresponding 3D target by overlaying the transformed shoe point cloud on the current observation.

to follow these value maps, which makes the method flexible for high-level spatial commands such as moving toward, away from, or above an object. However, its geometric grounding is sparse and hand-specified through code-level primitives. It struggles when the task requires precise continuous relations such as object-to-object rotations, contact surfaces, or fine insertion geometry.

ReKep (Huang et al., 2025) represents manipulation through VLM-predicted relational keypoints. It asks a VLM to identify task-relevant points and formulates trajectory optimization objectives over these keypoints, such as maintaining distances, alignments, or ordering constraints. This representation is interpretable, but it reduces object geometry to a small set of sparse points. The resulting constraints are sensitive to viewpoint, occlusion, and local depth noise, especially when the selected keypoints do not lie on stable or visible surfaces.

CoPA (Huang et al., 2024) uses VLMs to infer interaction constraints from visual prompts, typically grounding the task through contact points, keypoints, and projected direction vectors. Compared with pure language-code methods, CoPA provides more explicit visual grounding for where and how objects should interact. Nevertheless, its output is still a sparse proxy rather than a dense 3D object-level transformation. For tasks requiring full-body alignment, rotational consistency, or precise contact geometry, these projected vectors can be ambiguous or inconsistent under noisy RGB-D observations. LAMP instead estimates a continuous $SE(3)$ transformation over the active object relative to the passive object, preserving dense geometric relationships throughout perception, registration, and planning.

### A.15 More Discussions on Limitations

Although LAMP provides an explicit 3D representation for open-world manipulation, several limitations remain.

**Dependence on Image Editing Quality.** LAMP relies on edited images to specify the target state, so editing artifacts can affect the extracted 3D prior. We observe three representative cases: 1) *appearance changes in simulation*, where simulated assets are edited into realistic objects; 2) *passive-object displacement under partial views*, where hallucinated regions shift the passive reference object; and 3) *relative object-size changes*, where active/passive scale inconsistency leads to 3D offsets. We mitigate these issues with prompt templates, multiple edited candidates, mask-overlap rejection, and scale-consistent registration, but automatic detection of all editing failures remains open.

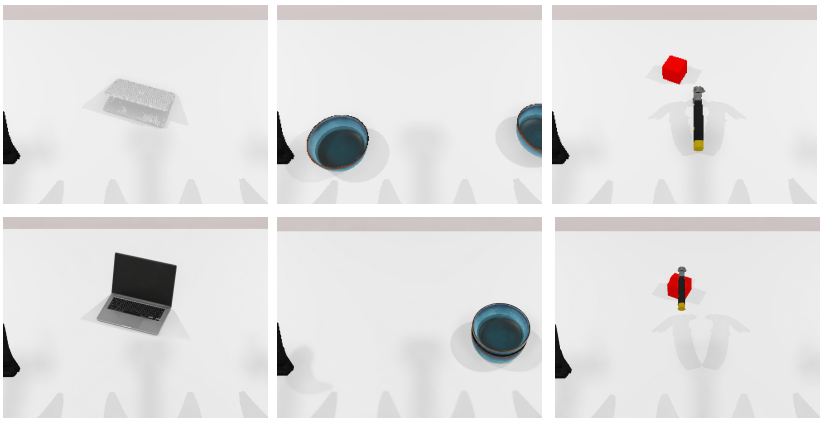


Figure 24: **Editing failure modes in simulation.** Top row: observed images. Bottom row: edited images. From left to right, the columns show appearance change in *Open Laptop*, passive/reference displacement in *Stack Bowls*, and relative size change in *Beat Block Hammer*.

**Segmentation and Registration Errors.** LAMP assumes that active and passive objects can be segmented and lifted into usable point clouds. Severe occlusion, thin objects, small parts, or single-view partial observations can leave incomplete geometry, and edited-state occlusion can further reduce the visible active-object point cloud. Although DINO-guided filtering and feature matching improve robustness, viewpoint or appearance changes may still bias correspondences. Moreover, the $SE(3)$ registration assumes near-rigid objects, which may not hold for highly deformable materials.

Table 7: **A list of 13 open-world manipulation tasks**. We provide the prompt used to generate the edited image in our experiment.

| | |
|---|---|
| *Egg placing* | move the egg onto the green holder |
| *Coin insertion* | insert the coin into the piggy bank |
| *Pencil insertion* | insert the pencil into the holder |
| *Toast insertion* | insert the toast into the toaster |
| *Lid covering* | move the lid onto the teapot |
| *Pen-cap covering* | cover the pen with the pen cap |
| *Tea pouring* | teapot pours into the cup |
| *Toast cutting* | cut the toast with the knife |
| *Block assembly* | move the green block near the blue block so that their jagged edges meet |
| *Ring stacking* | toss the red ring over the base |
| *Drawer opening* | pull out the red drawer |
| *Drawer closing* | push the red drawer in |
| *Toaster opening* | move the slider of the toaster downwards |

**Planning, Reachability, and Runtime.** The predicted transformation specifies the target object pose, but execution still depends on feasible grasps, robot reachability, and collision-free motion planning. If all valid grasps are occluded, the target lies outside the workspace, or the task requires intermediate dynamics not captured by a final pose, LAMP may fail despite a correct 3D estimate. The pipeline also follows a think-before-act design, so image editing and perception latency limit its use in highly dynamic scenes.

### A.16 Prompt for Image-Editing

We use Qwen-Image-Edit, Gemini 2.5 Flash Image (Nano Banana) and GPT-Image-2 as our editing models. The prompts used for each task in open-world manipulation are provided in Tab. 7.

### A.17 Implementation Details for Motion Planning

To implement a collision-aware motion planner, the observed environment is discretized into a voxel map. Once the robot grasps the object, the corresponding point cloud is approximated as a set of collision spheres. These spheres are then rigidly attached to the Tool Center Point (TCP) mesh, enabling efficient trajectory optimization that accounts for the object's geometry, as illustrated in Fig. 25.

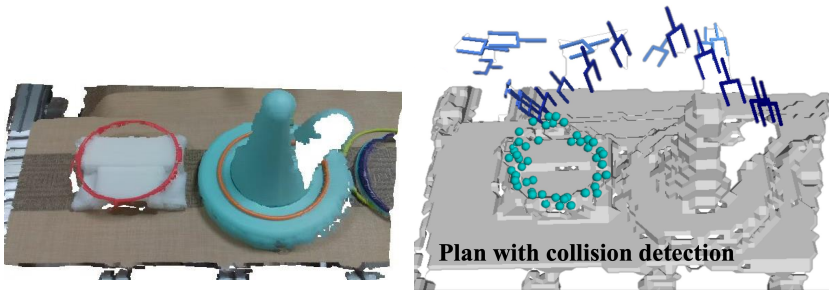



Figure 25: **Motion planning with collision detection.** We transform the observed environment (left) into a voxelized occupancy grid (gray, right). The grasped object is modeled as a spherical collision proxy (green spheres) integrated with the gripper. This representation enables the optimization of the robot's trajectory (blue) while maintaining safety constraints.