GOVLA: Image-Generative VLMs as Object-Centric World Models Enable Zero-shot Robot Manipulation

Haonan Chen*[1], Jingxiang Guo*[1], Bangjun Wang[2], Tianrui Zhang[1], Xuchuan Huang[3],
Boren Zheng[4], Yiwen Hou[1], Chenrui Tie[1], Jiajun Deng[1], Lin Shao†[1]

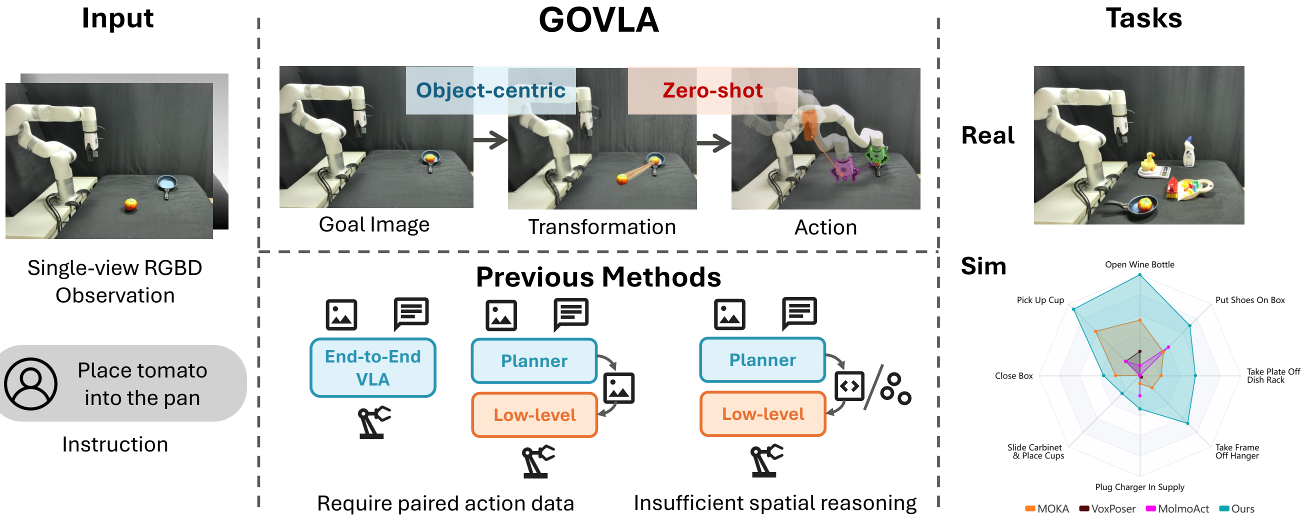

Fig. 1: **GOVLA** maps a single-view RGB-D image and a language instruction to executable manipulation actions. Our approach employs an object-centric world model to generate a goal image, from which the corresponding object transformation is subsequently computed, enabling zero-shot manipulation. This overcomes key limitations of previous methods, which often either rely on paired action data for training or lack precise spatial reasoning. We demonstrate our framework's strong performance and generalization capabilities across both simulated and real-world experiments.

***Abstract*— Generalization remains a fundamental challenge in robotic manipulation. To tackle this challenge, recent Vision-Language-Action (VLA) models build policies on top of Vision-Language Models (VLMs), seeking to transfer their open-world semantic knowledge. However, their zero-shot capability lags significantly behind the base VLMs, as the instruction-vision-action data is too limited to cover diverse scenarios, tasks, and robot embodiments. In this work, we present GOVLA, a zero-shot framework that leverages Image-Generative VLMs as world models to generate desired goal states, from which the target object pose is derived to enable generalizable manipulation. The key insight is that object state representation is the golden interface, naturally separating a manipulation system into high-level and low-level policies. This representation abstracts away explicit action annotations, allowing the use of highly generalizable VLMs while simultaneously providing spatial cues for training-free low-level control. To further improve robustness, we introduce a Reflection-through-Synthesis process that iteratively validates and refines the generated goal image before execution. Both simulated and real-world experiments demonstrate that our GOVLA achieves strong performance and inspiring generalizability in manipulation tasks. Supplementary materials are available at TBD .**

I. Introduction

Robotic manipulation is a long-standing topic for robotics and embodied AI [1]–[3]. It serves as a cornerstone for a wide range of applications in robotics, from household assistance and healthcare to manufacturing and logistics. Despite recent progress [4], methods that perform exceptionally well in specific settings often collapse under minor distributional shifts [5], [6] across environments, tasks, object categories, and robot embodiments. This fact makes robust generalization a valid problem and draws increasing research interest from the community. This generalization gap is the primary barrier hindering the practical deployment of autonomous robots in unstructured environments.

Foundation models, pre-trained on vast datasets, have emerged as a promising direction to address this challenge. This progress has catalyzed the development of Vision-Language-Action (VLA) models, a broad paradigm aiming to connect multimodal perception with robotic action. Two primary architectural approaches have emerged within the VLA paradigm: end-to-end and hierarchical.

The end-to-end VLA approach is to train a single, large-scale policy to map visual and language inputs to low-level robot actions directly. These models are either developed by finetuning existing Vision-Language Models (VLMs) [7]–[9]

* denotes equal contribution; † denotes the corresponding author.

[1]School of Computing, National University of Singapore; [2]The HKU Musketeers Foundation Institute of Data Science, The University of Hong Kong; [3]Yuanpei College, Peking University; [4]Department of Automation, Tsinghua University.

or by training high-capacity policies from the ground up on robotics data [10], [11]. While their architectures differ, both approaches share a common and significant challenge: their performance is contingent on massive paired instruction-vision-action data. This data type is exceptionally costly and time-consuming to acquire in sufficient diversity [12], [13], which limits its generalization to novel tasks and scenarios.

The hierarchical VLA approach, in contrast, aims to mitigate this intense data dependency. It decouples the problem using a VLM for high-level planning to generate an intermediate representation, guiding a separate low-level policy [14]–[16]. Yet, the design of the intermediate representation presents a critical dilemma. Sparse or symbolic representations, such as language descriptions and keypoints [14], [17]–[19], lack the precise geometric detail required for complex manipulation. In contrast, while dense visual representations like goal images provide rich information [20]–[23], they typically require the low-level policy to be explicitly trained on paired action data to interpret them, thus forfeiting the zero-shot objective. Other VLM-generated guidance, like value maps, constraints, or reward functions [15], [16], [24], often suffers from significant inaccuracies, stemming from the inherent weakness of current VLMs in precise spatial reasoning.

To ameliorate this problem, we develop our approach based on this key insight: Effective zero-shot generalization in a low-level policy requires explicit spatial guidance. However, the powerful VLMs serving as our high-level goal generator excel at semantic reasoning but perform worse at precise spatial reasoning. To this end, we propose a decoupled architecture that leverages the VLM as an object-centric world model. Unlike traditional agent-centric world models [22], [25]–[27] that are tied to specific robot kinematics and thus limit cross-embodiment generalization, our world model focuses exclusively on the semantic goal in the image space. This allows a separate, dedicated spatial grounding module to handle the translation of this semantic goal into explicit spatial guidance of the object, playing to the strengths of each component.

Formally, we introduce GOVLA, a zero-shot manipulation framework that employs an Image-Generative VLM to produce a goal image depicting the desired task outcome, functioning as an object-centric world model. This visual representation is translated into a precise target pose to direct a low-level policy. To evaluate and optimize the generated goal image, we incorporate a Reflection-through-Synthesis mechanism that assesses and iteratively refines the visual output. The target object state is then extracted from the refined goal image, and the goal pose is computed using feature matching and point cloud registration. This object-centric representation is inherently generalizable across various tasks, environments, object categories, and robot embodiments. By providing an explicit object pose rich in spatial information, our framework effectively guides a low-level policy to perform manipulation tasks in a zero-shot manner.

Our framework operates solely on a user's task description and a single-view RGB-D input, requiring no prior scene knowledge such as pre-scanned maps or object meshes. We demonstrate its effectiveness on tasks in simulated and real-world settings, including *Pick and Place, Table Sweeping, Bottle Stand-Up, and Box Closing*. Our approach consistently outperforms baseline methods, such as MOKA [19], VoxPoser [15], and MolmoAct [9], achieving higher success rates. Moreover, our pipeline does not require task-specific fine-tuning, highlighting its zero-shot generalization.

To summarize, our key contributions are:

- We introduce GOVLA, a decoupled hierarchical framework that leverages an Image-Generative VLM as a world model to generate goal object states in image form, serving as the bridge between high-level semantic reasoning and low-level action control.
- We propose an iterative refinement process, termed Reflection-through-Synthesis, where the generated goal is improved by synthesizing and overlaying a virtual image of the target object onto the current observation for visual inspection and improvement.
- We demonstrate that GOVLA achieves inspiring zero-shot generalizability across diverse manipulation tasks, environments, object categories, and robot embodiments in both simulation and the real world, without requiring any task-specific finetuning.

II. Related Work

TABLE I: Comparison of VLA Paradigms

Category	Method	Future State Awareness	Inter-mediate Type	Training-Free Policy	Precise 3D Spatial Grounding
End-to-End VLA	RT-1/2 [10], [11], OpenVLA [7], π_0 [28], PaLM-E [18]	×	N/A	×	×
	MolmoAct [9]	×	Depth & Trajectory	×	✓
Hierarchical VLA	SayCan [14]	×	Symbolic Skills	×	×
	Code as Policies [17]	×	Code	×	×
	ProgPrompt [29]	×	Programmatic Plan	×	×
	MOKA [19]	✓	Keypoints	✓	×
	Rekep [16]	×	Keypoints	✓	×
	IKER [24]	✓	Rewards	×	×
	SUSIE [20], Stone et al. [23]	✓	Subgoal Images	×	×
	VoxPoser [15]	×	3D Value Maps	✓	×
	3D-VLA [21]	✓	3D Scene	×	✓
	GOVLA (Ours)	✓	Object State	✓	✓

A. *Foundation Models Paradigms for Robotic Manipulation*

Recent advances in foundation models have significantly influenced robotics, especially in robotic manipulation tasks, by leveraging LLMs and VLMs for high-level planning and decision-making [30]–[33]. A key development in this area is the Vision-Language-Action (VLA) model: a class of frameworks that maps multimodal inputs, typically visual observations and language instructions, to executable robot actions [31], [33]. Current research building on this concept can be broadly categorized into several main paradigms, which we compare in detail in Table I.

End-to-End Vision-Language-Action Models. The first paradigm aims to build monolithic, end-to-end models that directly map visual and linguistic inputs to low-level robot actions. This approach includes methods that train large models from scratch, such as RT-1 [10] and RT-2 [11], as well as those that finetune pre-trained VLMs, like OpenVLA [7], RDT-1B [8], and MolmoAct [9]. While these models have impressive capabilities, their performance is fundamentally bottlenecked by the need for massive datasets of paired instruction-vision-action data. The high cost and complexity of collecting such data severely constrain their diversity, limiting the models' zero-shot generalization.

Hierarchical Models with Intermediate Representations. A second representative paradigm employs foundation models in a hierarchical structure. A high-level VLM is a planner in this setup, generating an intermediate representation to guide a separate low-level policy [34]. Therefore, the design of this intermediate representation is critical, defining the primary trade-offs within this paradigm. One line of work utilizes symbolic or sparse geometric representations. This includes selecting from a library of predefined skills [14], [18], generating executable code [17], [29], or defining tasks through a set of keypoints [19], [35]. While effective for structured tasks, the generalizability of these methods is often constrained by the limited scope of discrete skill sets or the insufficiency of sparse keypoints to capture complex object interactions. Other methods generate more explicit, dense visual guidance to provide richer context, such as subgoal images [20], [21] or videos [22], [36]. A significant drawback, however, is that the low-level policy typically needs to be trained to interpret these visual goals, which reintroduces a data dependency and undermines the objective of a truly zero-shot, training-free system. A third approach involves the VLM generating implicit spatial guidance, like value maps [15], constraints [16], or reward functions [24]. These methods require the VLM to ground its reasoning in 3D space, a capability that is often imprecise due to the inherent limitations of its spatial understanding.

B. Reflection in Foundation Models

Reflection mechanisms, which enable generative models to iteratively critique and refine their outputs, have recently attracted growing attention as a promising approach for enhancing robotic manipulation capabilities. In generative modeling, recent studies demonstrate that incorporating self-feedback or iterative critiques substantially improves the quality and coherence of generated outputs [37]–[39]. Notable examples include CritiqueLLM [40] and Idea2Img [41], which showcase how reflective feedback loops facilitate progressive refinement and correction of initial predictions. Extending these reflective approaches into robotics, several recent frameworks [42]–[44] integrate self-reflection into robotic task planning and action execution, enabling robotic agents to dynamically identify and correct errors, thereby progressively enhancing their performance during tasks. Moreover, additional studies have advanced this concept by incorporating multimodal reflection mechanisms, effectively bridging high-level cognitive reasoning with low-level motor control adjustments. This multimodal integration significantly improves robot robustness and adaptability, enabling robots to manage uncertainties better and effectively generalize across diverse manipulation scenarios and real-world conditions [45]–[47].

III. Method

The overall workflow of our framework is illustrated conceptually in Figure 2 and detailed procedurally in Algorithm 1. This section begins by formulating the problem and defining the inputs and outputs for each module of our hierarchical pipeline (Sec. III-A). We then detail the core components, starting with the Goal State Reasoning module, which interprets the user's instruction to generate a visual goal state (Sec. III-B). Subsequently, the Spatial Grounding module takes this visual representation and computes a precise 3D transformation (Sec. III-C). Finally, we describe how the Low-level Policy uses this transformation to plan and execute the physical manipulation task (Sec. III-D).

A. Problem Formulation

Given a single-view RGBD image observation $O = (\mathcal{I} \in \mathbb{R}^{H\times W\times 3}, \mathcal{D} \in \mathbb{R}^{H\times W\times 1})$, and a natural language task description $\mathcal{L}$, the objective is to generate an action sequence $\{\boldsymbol{a}\}_i$ that completes the manipulation task described in $\mathcal{L}$.

Algorithm 1 GOVLA Execution Framework

```
Require: Initial observation $O = (\mathcal{I}, \mathcal{D})$, Language instruction $\mathcal{L}$, Initial End-effector pose $P_{init}$
Ensure: Action sequence $\{\boldsymbol{a}\}_i$
procedure GOVLA($O, \mathcal{L}$)
Stage 1: Goal State Reasoning
    $\mathcal{L}_e \leftarrow$ EnhancePrompt($\mathcal{I}, \mathcal{L}$)
# — Reflection-through-Synthesis Loop —
    for $i = 1$ to max_iterations do
        $\mathcal{I}'_{cand} \leftarrow$ GenerateImage($\mathcal{I}, \mathcal{L}_e$)
        $\mathcal{I}_{synth} \leftarrow$ SynthesizeOverlay($\mathcal{I}, \mathcal{I}'_{cand}$)
        is_valid, $\mathcal{L}_{revised} \leftarrow$ Reflector($\mathcal{I}_{synth}, \mathcal{L}_e$)
        if is_valid then break
        end if
        $\mathcal{L}_e \leftarrow \mathcal{L}_{revised}$
    end for
    $\mathcal{I}', \mathcal{M}, \mathcal{M}', \mathcal{D}' \leftarrow$ GenMask&Depth($\mathcal{I}, \mathcal{I}'_{cand}$)
Stage 2: Spatial Grounding
    $(R, t) \leftarrow$ ComputeTransform($\mathcal{I}, \mathcal{I}', \mathcal{D}, \mathcal{D}', \mathcal{M}, \mathcal{M}'$)
Stage 3: Low-level Policy
    $P_{contact} \leftarrow$ DetermineContactPose($O, \mathcal{M}$)
    $P_{goal} \leftarrow$ ApplyTransform($(R, t), P_{contact}$)
    $\{\boldsymbol{a}\}_i \leftarrow$ MotionPlan($P_{init}, P_{contact}, P_{goal}$)
    return $\{\boldsymbol{a}\}_i$
end procedure
```

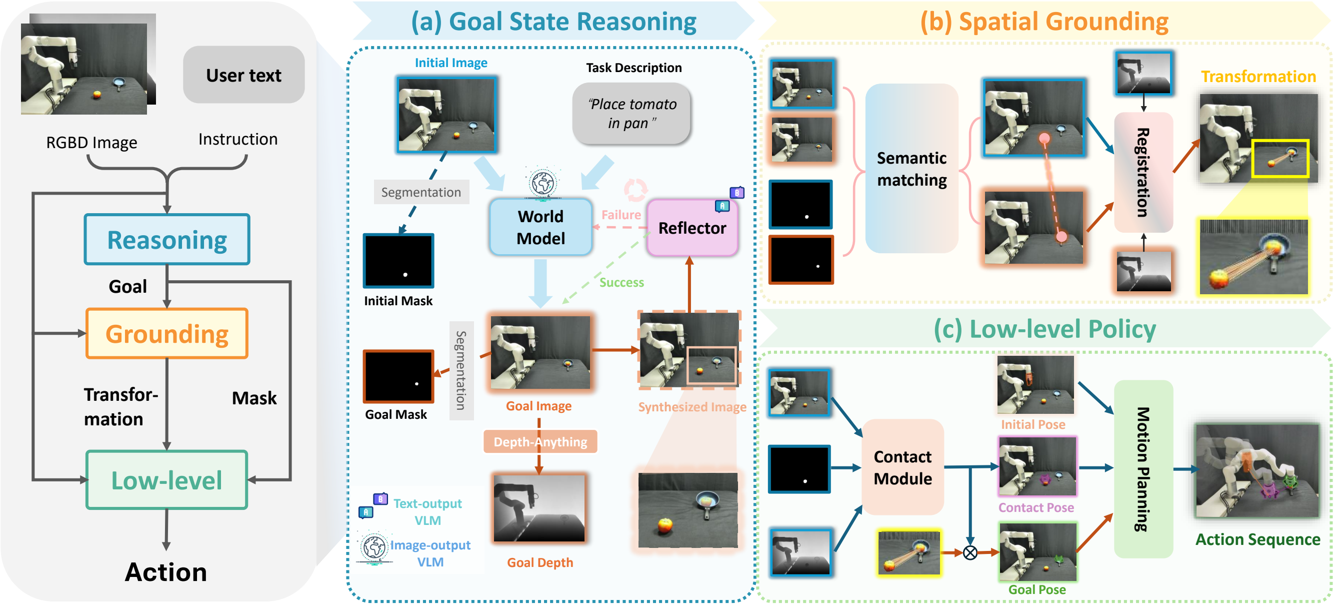

Fig. 2: Overview of the **GOVLA** framework, which decouples the manipulation pipeline into three stages: **(a) Goal State Reasoning:** A VLM generates a goal image from instructions and refines it for task feasibility, yielding a validated goal with image, mask, and depth. **(b) Spatial Grounding:** The object's transformation is computed by feature matching and point cloud registration between the initial and goal states. **(c) Low-level Policy:** The gripper's goal pose is derived by applying the object's transformation to a contact pose, after which a motion planner generates the final trajectory for robot execution.

The **Goal State Reasoning** receives an RGB image $\mathcal{I}$ and a task description $\mathcal{L}$ to produce a goal image $\mathcal{I}' \in \mathbb{R}^{H\times W\times 3}$ and a goal metric depth map $\mathcal{D}' \in \mathbb{R}^{H\times W\times 1}$. The Goal State Reasoning module also generates the object masks for the initial and goal images, denoted as $\mathcal{M}$ and $\mathcal{M}'$.

The **Spatial Grounding** is given with the initial and goal RGB images $\mathcal{I}, \mathcal{I}'$, real depth maps $\mathcal{D}$ and goal metric depth $\mathcal{D}'$, and object masks $\mathcal{M}, \mathcal{M}'$. The objective of this module is to compute the rotation $R \in \mathbb{R}^{3\times 3}$ and translation $t \in \mathbb{R}^3$ that maps the object from its initial pose to its goal pose.

The **Low-level Policy** takes the current observation $\mathcal{O} = (\mathcal{I}, \mathcal{D})$ and the mask $\mathcal{M}$ as input, then outputs a sequence of actions $\{\boldsymbol{a}\}_i$ to drive the system from $\mathcal{O}$ to $\mathcal{O}' = (\mathcal{I}', D')$.

B. Goal State Reasoning

The Goal State Reasoning module is responsible for translating the user's abstract language instruction ($\mathcal{L}$) into a concrete and plausible visual goal, represented by a goal image ($\mathcal{I}'$) and a goal depth map ($\mathcal{D}'$). This is achieved through an initial prompt enhancement step and an iterative refinement loop we term Reflection-through-Synthesis.

Before the image generation process, we enhance the user's raw instruction to make it more descriptive for the image generation model. The initial image $\mathcal{I}$ and the raw task description $\mathcal{L}$ are fed into a text-output VLM (e.g., Gemini 2.5 Pro). This model enriches the concise instruction with critical details inferred from the visual context, producing a more descriptive enhanced prompt.

The Reflection-through-Synthesis Loop: This enhanced prompt then seeds our iterative refinement loop, which generates and validates candidate goals until one is deemed successful. Each iteration involves the following steps:

- **Goal Image Generation:** The current prompt and the initial image $\mathcal{I}$ are passed to a pre-trained image generation VLM (Gemini 2.5 Flash-image, also known as Nano Banana) to produce a candidate goal image.
- **Synthesis for Inspection:** We create a composite image for the reflector to facilitate evaluation. Using Grounded SAM [48], we segment the target object from the candidate goal and overlay this "virtual image" onto the initial scene with partial transparency.
- **Reflection and Refinement:** This synthesized image, along with the original instruction and the current enhanced prompt, is fed to a **Reflector VLM** (Gemini 2.5 Pro). The Reflector assesses if the synthesized outcome aligns with the task's semantic requirements. If successful, the loop terminates. If not, the Reflector generates a revised prompt containing corrective feedback, which is used for the next generation iteration.

This process continues until the Reflector validates a goal image or a pre-set maximum number of attempts is reached. Once a goal image is accepted, the final outputs for the next module are generated: a depth estimation model (Depth-Anything V2 [49]) produces the goal's relative depth map. At the same time, Grounded SAM [48] extracts the target object masks for both the initial and final goal images.

C. Spatial Grounding

The spatial transformation module consists of two main stages: **semantic matching** and **point-cloud registration**.

Semantic Matching. Given the initial observation $\mathcal{I} \in \mathbb{R}^{H\times W\times 3}$ and the goal image $\mathcal{I}' \in \mathbb{R}^{H\times W\times 3}$, our goal is to obtain a pixel matching $\mathcal{F} : (x, y) \to (x', y')$, which maps pixel in the intial frame $\mathcal{I}$ to the corresponding pixel in the generated frame $\mathcal{I}'$. Due to the inconsistency introduced by our generative paradigm, pixel matching cannot be reached by traditional flow estimation models or 2D pixel tracking models. Instead, we rely on semantic features to generate robust pixel matching. Based on the existing semantic matching model Geo-Aware [50], we extract pixel-wise semantic features $\mathbf{f}, \mathbf{f}'$ from $\mathcal{I}, \mathcal{I}'$, respectively. Furthermore, we match pixels between the two frames by pairing the pixel in $\mathcal{I}$ with the pixel yielding the highest cosine similarity among all pixels in $\mathcal{I}'$, i.e.

$$(x', y') = \arg\max_{(p,q)} \frac{\mathbf{f}_{(x,y)} \cdot \mathbf{f}'_{(p,q)}}{\|\mathbf{f}_{(x,y)}\|\|\mathbf{f}'_{(p,q)}\|} \tag{1}$$

Point-cloud Registration. Following the establishment of 2D correspondences, this stage computes the 3D spatial transformation. The process involves lifting the 2D images to 3D point clouds using depth information. While a ground-truth depth map $\mathcal{D}$ is available for the initial frame, the depth map for the generated goal, $\mathcal{D}'$, is estimated using the DepthAnythingV2 model, which introduces scale ambiguity.

To resolve this ambiguity, we first perform depth alignment. A linear transformation, characterized by a scale factor s_1 and a bias b, is calibrated to map the predicted depth scale of $\mathcal{D}'$ to the metric scale of $\mathcal{D}$. This transformation is determined via the method of least squares:

$$\mathcal{D}[(\mathcal{M} \cup \mathcal{M}')^c] = s_1 \cdot \mathcal{D}'[(\mathcal{M} \cup \mathcal{M}')^c] + b, \tag{2}$$

The regression is performed exclusively on background pixels, denoted by the complement set $(\mathcal{M} \cup \mathcal{M}')^c$, to ensure manipulated foreground objects do not skew the alignment. The learned parameters are then applied to the entire predicted depth map to yield a correctly scaled version,

$$\mathcal{D}'_{\text{rescaled}} = s_1 \cdot D' + b \tag{3}$$

Subsequently, object-specific point clouds, $\mathcal{P}$ and $\mathcal{P}'$, are extracted from the initial and goal frames by applying object masks $\mathcal{M}$ and $\mathcal{M}'$ to the depth maps $\mathcal{D}$ and $\mathcal{D}'_{\text{rescaled}}$, respectively. To account for potential changes in object size introduced by the Image-Generative VLM, we solve for a similarity transformation between the two point clouds:

$$s_2 \cdot \mathcal{P}' = R\mathcal{P} + t, \tag{4}$$

where R stands for the rotation; t stands for translation and s_2 is the scale factor of the transformation, respectively. The optimal rotation R and translation t are computed using the Kabsch algorithm [51]. These final parameters, representing the precise spatial transformation, are then utilized by a low-level policy module to guide robot actions.

D. Low-level Policy

The **Low-level Policy** translates the abstract object goal pose, provided by the high-level modules, into executable robot actions. This process involves three stages: determining an initial contact point on the object, computing the gripper's final goal pose, and planning a valid trajectory. A key advantage of our framework's object-centric representation is its flexibility; it provides an explicit geometric target and is compatible with various low-level controllers operating on different input modalities (e.g., RGB images or point clouds).

Contact Module. The Contact Module determines a feasible contact pose using a multi-stage, sample-based method on the object's point cloud. First, a large set of grasp candidates is generated on the object's surface, oriented along local surface normals. This set is then pruned through filters that check for valid orientation and perform collision checks using a 3D gripper mesh against the scene. Finally, the remaining collision-free candidates are scored by the number of object points within the gripper's closing volume, and the top-scoring grasp is selected as the contact pose for subsequent planning.

Goal Pose Computing. We assume that the relative pose between the gripper and the object remains constant after contact. This assumption holds for most prehensile actions and a subset of non-prehensile tasks. Based on this, we calculate the gripper's goal pose by applying the object's transformation, as determined by the Spatial Grounding module, to the gripper's initial pose. This provides the target configuration for motion planning.

Motion Planning Module. Finally, the Motion Planning Module generates a collision-free trajectory for the manipulator from its current configuration to the target pose. In our simulated experiments, we leverage a built-in sample-based planner to interpolate the trajectory. We utilized a motion planner in the experiment to generate a feasible path in the robot's configuration space.

IV. Experiment

In this section, we conduct comprehensive experiments and analyses to answer the following key questions:

Q1: How well does our proposed method perform compared to existing baselines?

Q2: How effective is the Reflection-through-Synthesis process in refining the outputs of the high-level reasoning?

Q3: Can our framework generalize across diverse environments, tasks, object categories, and robot embodiments?

A. Experimental Setup

Simulation. Our simulation experiments were conducted in RLBench [52], utilizing a Franka Emika Panda 7-DoF arm with attached RGB-D cameras. The robot arm is fixed to a tabletop, and for each task, objects are placed in randomized initial configurations determined by a random seed. We selected eight representative tasks from the RLBench benchmark to evaluate a range of manipulation skills, with the complete list and setup details provided in Table II. In

TABLE II: Results of Simulation Experiments. All methods are evaluated in a zero-shot setting.

Method	Pick Up Cup	Open Wine Bottle	Put Shoes On Box	Take Plate Off Dish Rack	Take Frame Off Hanger	Plug Charger In Supply	Slide Cabinet & Place Cups	Close Box	Average Success Rate
OpenVLA	0/100	2/100	0/100	0/100	0/100	0/100	0/100	0/100	0.2%
Pi0	0/100	0/100	0/100	0/100	0/100	0/100	0/100	0/100	0.0%
SUSIE	0/100	0/100	0/100	0/100	0/100	0/100	0/100	0/100	0.0%
MOKA	62/100	55/100	33/100	21/100	17/100	8/100	0/100	24/100	26.0%
VoxPoser	20/100	24/100	0/100	0/100	2/100	0/100	0/100	0/100	5.8%
MomolAct	20/100	10/100	40/100	0/100	0/100	20/100	0/100	0/100	11.3%
Ours	93/100	100/100	70/100	55/100	67/100	33/100	25/100	36/100	59.9%

all experiments, the robot starts without holding any object at the beginning of each trial.

To robustly assess performance and account for variations in object placement, each task was evaluated across 10 random seeds. For each seed, which defines a unique initial scene arrangement, we conducted 10 independent trials of each method, resulting in 100 evaluation runs per task for calculating the success rate.

Real World. Our real-world experimental setup comprises a 7-DoF UFACTORY X-ARM 7 and an Orbbec Femto Bolt RGB-D camera. We designed four distinct real-world tasks to evaluate a range of core manipulation capabilities: *Place Tomato in Pan*, a foundational pick-and-place task requiring reasoning about object containment; *Table Sweeping*, which involves tool use and non-prehensile manipulation; *Weigh the Duck*, a task demanding precise placement onto a small target; and *Stand a Bottle Upright*, which tests object reorientation. We conducted 10 trials for each task, with detailed results presented in Table III and Figure 5.

B. Baselines

We evaluate our method against representative baselines from the two primary paradigms discussed in our related work: end-to-end and hierarchical. Our selection is designed to rigorously test our approach against the main competing strategies in robotic manipulation.

For the end-to-end paradigm, we select OpenVLA [7], Pi0 [28], and MolmoAct [9]. These models are typical examples of the end-to-end VLA approach that directly maps multimodal inputs to robot actions.

We choose a diverse set of methods for the hierarchical paradigm to evaluate the efficacy of different intermediate representations—the key design trade-off in this category. Our selection includes: VoxPoser [15] (value maps), SUSIE [20] (subgoal images), and MOKA [19] (keypoints), covering a broad spectrum of representations, from implicit spatial guidance to explicit visual and geometric targets.

All of our comparative evaluations are conducted under a strict zero-shot protocol, which means neither our method nor the baselines undergo any task-specific fine-tuning.

C. Simulation Experiments (Q1)

The results of our simulation experiments are presented in Table II. Our method, **GOVLA**, achieves a remarkable average success rate of 59.9%, significantly outperforming all baselines across a diverse set of eight manipulation tasks.

The end-to-end baselines, OpenVLA and Pi0, failed almost completely (0.2% and 0% success rates, respectively), highlighting their brittleness in this zero-shot setting due to a strong dependency on large-scale, in-domain action data. Performance within the hierarchical paradigm was varied but limited by the intermediate representation. While methods using keypoints (MOKA, 26.0%) or trajectories (MomolAct, 11.3%) showed some capability, they were still substantially outperformed by our approach. This suggests their representations, unlike our explicit 3D pose, lack sufficient spatial precision or require task-specific training for the low-level policy to be effective.

These findings decisively answer our first research question (Q1). While other methods falter due to data dependency or imprecise goals, GOVLA's explicit goal object state provides the necessary precision and robustness for a low-level policy to succeed.

D. Ablation Study (Q2)

We perform an ablation study to validate the contributions of our two key components: Input Enhancement and the Reflection-through-Synthesis process.

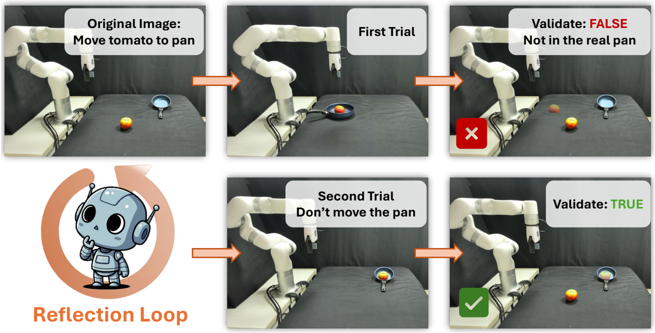

Fig. 3: An example of our **Reflection-through-Synthesis** process, which corrects a semantically correct but infeasible goal by refining the generation prompt.

Reflection's Necessary: Figure 3 highlights a typical failure mode of image generation. Image-Generative VLM may alter non-target elements (e.g., moving the pan along with the tomato), creating a semantically plausible but physically incorrect goal. Our **synthesis** step, overlaying the segmented target object onto the original scene, grounds the reflection in the world context. This allows our Reflector VLM to identify such errors and provide corrective feedback effectively.

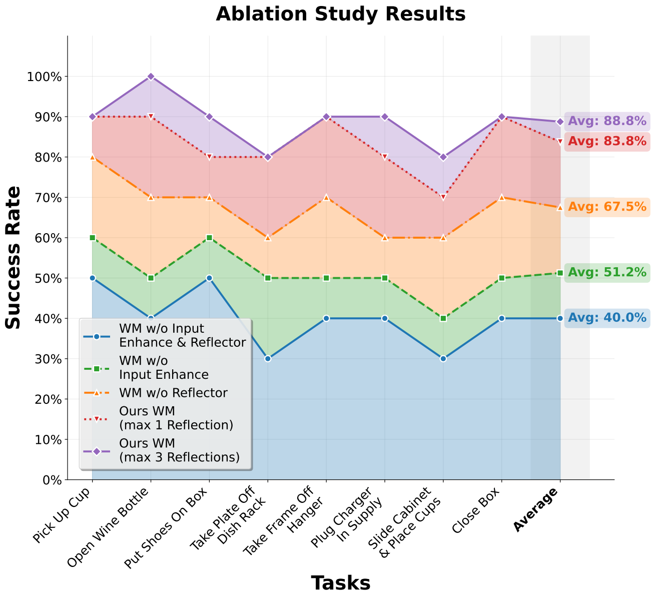

Fig. 4: Ablation Study. The performance of our full model ("World Model w/ Instruction & max 3 Reflection"), shown by the purple line, surpasses all ablated variants.

Quantitative Analysis: The results in Figure 4 quantify the impact of each component. Starting from a 40.0% success rate for the baseline model, adding Input Enhancement provides the most significant single improvement (+27.5pp), while the Reflector alone yields a substantial gain (+11.2pp). Our complete model, which combines both, performs best, and its success can be further boosted from 83.8% to 88.8% by increasing the maximum reflection iterations from 1 to 3.

These results demonstrate that both components are critical and complementary, confirming their effectiveness and answering our second research question (Q2).

E. Real World Experiments (Q3)

Our framework was tested on four diverse real-world manipulation tasks to validate its practical applicability. As shown in Table III, our method achieved a 60% average success rate, significantly outperforming hierarchical baselines like MOKA (22.5%) and MolmoAct (27.5%). Consistent with the simulation findings, the end-to-end model OpenVLA failed (0%), further highlighting our object-centric representation's robustness.

TABLE III: Real World Experiments

Method	Tomato Placement	Table Sweeping	Weighing Duck	Bottle Stand-Up	Average Success Rate
OpenVLA	0/10	0/10	0/10	0/10	0%
MOKA	5/10	1/10	3/10	0/10	22.5%
MolmoAct	5/10	0/10	6/10	0/10	27.5%
Ours	9/10	4/10	7/10	4/10	60%

Figure 5 provides qualitative evidence for these findings. The visualizations illustrate how the generated goal image captures the task's intent, and the subsequent transformation provides a precise spatial target, enabling the robot to execute complex tasks in a robust, zero-shot manner.

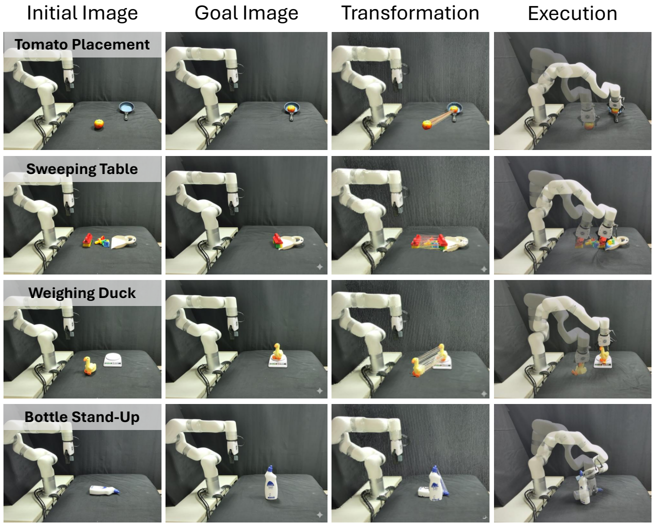

Fig. 5: Qualitative Results from Real-World Experiments.

The framework's success across diverse tasks, objects, and environments (simulated and real), combined with its zero-shot deployment on different robot embodiments, demonstrates strong generalization, providing a clear answer to Q3.

F. Failure Cases Analysis

In our real-world experiments, we observed several typical failure modes as different tasks placed varying demands on each module of our framework.

Failures originating from the Spatial Grounding module were the primary obstacle in several precision-demanding tasks. For *Tomato Placement* and *Weighing Duck*, this manifested as inaccuracies in the estimated depth, with the latter task being sensitive to minor height errors. Similarly, the *Bottle Stand-Up* task's success was contingent on orientation precision, where even minor inaccuracies in the computed orientation from this module would lead to failure.

In contrast, the primary difficulty in the *Table Sweeping* task shifted to the High-level Reasoning module. Generating a goal image representing a practical and feasible sweeping motion demands a sophisticated semantic understanding, and occasional failures occurred at this reasoning stage.

V. Conclusion

In this work, we introduced **GOVLA**, a hierarchical manipulation framework designed to address the critical challenge of zero-shot generalization. Our key contribution is a decoupled architecture where a high-level generative VLM, acting as an object-centric world model, envisions a visual goal state. This visual goal is iteratively refined for plausibility and accuracy through our novel Reflection-through-Synthesis mechanism. Crucially, this visual representation is translated into a precise 3D object pose, which serves as an explicit, training-free guide for a separate low-level policy. This design ensures the entire manipulation process, from semantic reasoning to physical execution, is performed in a zero-shot setting, eliminating any requirement for action data or task-specific finetuning. Our extensive evaluations in both simulation and the real world demonstrate that GOVLA significantly surpasses existing end-to-end and hierarchical baselines across various manipulation tasks, highlighting its strong generalization and cross-embodiment capabilities.

References

[1] Z. Xu, K. Wu, J. Wen, J. Li, N. Liu, Z. Che, and J. Tang, "A survey on robotics with foundation models: toward embodied ai," *arXiv preprint arXiv:2402.02385*, 2024. 1

[2] Y. Liu, W. Chen, Y. Bai, X. Liang, G. Li, W. Gao, and L. Lin, "Aligning cyber space with physical world: A comprehensive survey on embodied ai," *arXiv preprint arXiv:2407.06886*, 2024. 1

[3] Z. Xian, T. Gervet, Z. Xu, Y.-L. Qiao, T.-H. Wang, and Y. Wang, "Towards generalist robots: A promising paradigm via generative simulation," *arXiv preprint arXiv:2305.10455*, 2023. 1

[4] R. Shao, W. Li, L. Zhang, R. Zhang, Z. Liu, R. Chen, and L. Nie, "Large vlm-based vision-language-action models for robotic manipulation: A survey," *arXiv preprint arXiv:2508.13073*, 2025. 1

[5] N. Roy, I. Posner, T. Barfoot, and et al., "From machine learning to robotics: Challenges and opportunities for embodied intelligence," *arXiv preprint arXiv:2110.15245*, 2021. 1

[6] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges," *Found. Trends Mach. Learn.*, 2019. 1

[7] M. J. Kim, K. Pertsch, S. Karamcheti, and et al., "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024. 1, 2, 3, 6

[8] S. Liu, L. Wu, B. Li, and et al., "Rdt-1b: a diffusion foundation model for bimanual manipulation," *arXiv:2410.07864*, 2024. 1, 3

[9] J. Lee, J. Duan, H. Fang, and et al., "Molmoact: Action reasoning models that can reason in space," *arXiv:2508.07917*, 2025. 1, 2, 3, 6

[10] A. Brohan, N. Brown, J. Carbajal, and et al., "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022. 2, 3

[11] A. Brohan, N. Brown, and J. C. et al., "Rt-2: Vision-language-action models transfer web knowledge to robotic control," 2023. [Online]. Available: https://arxiv.org/abs/2307.15818 2, 3

[12] J. Yang, W. Tan, C. Jin, and et al., "Transferring foundation models for generalizable robotic manipulation," in *WACV*. IEEE, 2025. 2

[13] A. O'Neill, A. Rehman, A. Maddukuri, and et al., "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *ICRA*, 2024, pp. 6892–6903. 2

[14] M. Ahn, A. Brohan, N. Brown, and et al., "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022. 2, 3

[15] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023. 2, 3, 6

[16] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, "Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation," *arXiv preprint arXiv:2409.01652*, 2024. 2, 3

[17] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *ICRA*. IEEE, 2023, pp. 9493–9500. 2, 3

[18] D. Driess, F. Xia, and M. S. M. e. a. Sajjadi, "Palm-e: An embodied multimodal language model," in *ICML*, vol. 202. PMLR, 2023, pp. 8469–8488. 2, 3

[19] F. Liu, K. Fang, P. Abbeel, and S. Levine, "Moka: Open-vocabulary robotic manipulation through mark-based visual prompting," in *ICRA Workshop*, 2024. 2, 3, 6

[20] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine, "Zero-shot robotic manipulation with pretrained image-editing diffusion models," *arXiv:2310.10639*, 2023. 2, 3, 6

[21] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan, "3d-vla: A 3d vision-language-action generative world model," *arXiv preprint arXiv:2403.09631*, 2024. 2, 3

[22] J. Wu, S. Yin, N. Feng, X. He, D. Li, J. Hao, and M. Long, "ivideogpt: Interactive videogpts are scalable world models," *NeurIPS*, vol. 37, pp. 68 082–68 119, 2024. 2, 3

[23] A. Stone, T. Xiao, and Y. e. a. Lu, "Open-world object manipulation using pre-trained vision-language models," *arXiv preprint arXiv:2303.00905*, 2023. 2

[24] S. Patel, X. Yin, W. Huang, S. Garg, H. Nayyeri, L. Fei-Fei, S. Lazebnik, and Y. Li, "A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards," *arXiv preprint arXiv:2502.08643*, 2025. 2, 3

[25] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019. 2

[26] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *CoRL*. PMLR, 2023, pp. 2226–2240. 2

[27] C. Zhu, R. Yu, S. Feng, B. Burchfiel, P. Shah, and A. Gupta, "Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets," *arXiv preprint arXiv:2504.02792*, 2025. 2

[28] K. Black, N. Brown, and D. D. et al., "π_0: A vision-language-action flow model for general robot control," 2024. [Online]. Available: https://arxiv.org/abs/2410.24164 2, 6

[29] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," *arXiv preprint arXiv:2209.11302*, 2022. 2, 3

[30] R. Firoozi, J. Tucker, S. Tian, and et al., "Foundation models in robotics: Applications, challenges, and the future," *IJRR*, p. 02783649241281508, 2023. 2

[31] Y. Ma, Z. Song, Y. Zhuang, J. Hao, and I. King, "A survey on vision-language-action models for embodied ai," *arXiv preprint arXiv:2405.14093*, 2024. 2, 3

[32] C. Tie, S. Sun, J. Zhu, and et al., "Manual2skill: Learning to read manuals and acquire robotic skills for furniture assembly using vision-language models," *arXiv preprint arXiv:2502.10090*, 2025. 2

[33] Y. Zhong, F. Bai, S. Cai, and et al., "A survey on vision-language-action models: An action tokenization perspective," *arXiv preprint arXiv:2507.01925*, 2025. 2, 3

[34] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *ICLR*, 2023. 3

[35] Y. Mikami, A. Melnik, J. Miura, and V. Hautamäki, "Natural language as policies: Reasoning for coordinate-level embodied control with llms," *arXiv preprint arXiv:2403.13801*, 2024. 3

[36] H. Bharadhwaj, D. Dwibedi, A. Gupta, and et al., "Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation," *arXiv preprint arXiv:2409.16283*, 2024. 3

[37] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," *NeurIPS*, vol. 36, pp. 8634–8652, 2023. 3

[38] A. Madaan, N. Tandon, P. Gupta, and et al., "Self-refine: Iterative refinement with self-feedback," *NeurIPS*, vol. 36, 2023. 3

[39] S. S. Raman, V. Cohen, I. Idrees, E. Rosen, R. Mooney, S. Tellex, and D. Paulius, "Cape: Corrective actions from precondition errors using large language models," in *ICRA*. IEEE, 2024, pp. 14 070–14 077. 3

[40] P. Ke, B. Wen, Z. Feng, and et al., "Critiquellm: Towards an informative critique generation model for evaluation of large language model generation," *arXiv preprint arXiv:2311.18702*, 2023. 3

[41] Z. Yang, J. Wang, L. Li, K. Lin, C.-C. Lin, Z. Liu, and L. Wang, "Idea2img: Iterative self-refinement with gpt-4v for automatic image design and generation," in *ECCV*. Springer, 2024, pp. 167–184. 3

[42] Z. Wang, S. Cai, G. Chen, A. Liu, X. Ma, and Y. Liang, "Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents," *arXiv preprint arXiv:2302.01560*, 2023. 3

[43] Y. Feng, J. Han, Z. Yang, X. Yue, S. Levine, and J. Luo, "Reflective planning: Vision-language models for multi-stage long-horizon robotic manipulation," *arXiv preprint arXiv:2502.16707*, 2025. 3

[44] Z. Yang, C. Garrett, D. Fox, T. Lozano-Pérez, and L. P. Kaelbling, "Guiding long-horizon task and motion planning with vision language models," *arXiv preprint arXiv:2410.02193*, 2024. 3

[45] J. Liu, C. Li, G. Wang, and et al., "Self-corrected multimodal large language model for robot manipulation and reflection." 3

[46] C. Xiong, C. Shen, X. Li, K. Zhou, J. Liu, R. Wang, and H. Dong, "Aic mllm: Autonomous interactive correction mllm for robust robotic manipulation," *arXiv preprint arXiv:2406.11548*, 2024. 3

[47] W. Xia, R. Feng, D. Wang, and D. Hu, "Phoenix: A motion-based self-reflection framework for fine-grained robotic action correction," *arXiv preprint arXiv:2504.14588*, 2025. 3

[48] T. Ren, S. Liu, A. Zeng, and et al., “Grounded sam: Assembling open-world models for diverse visual tasks,” *arXiv preprint arXiv:2401.14159*, 2024. 4

[49] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, “Depth anything: Unleashing the power of large-scale unlabeled data,” in *CVPR*, 2024, pp. 10 371–10 381. 4

[50] J. Zhang, C. Herrmann, J. Hur, E. Chen, V. Jampani, D. Sun, and M.-H. Yang, “Telling left from right: Identifying geometry-aware semantic correspondence,” in *CVPR*, 2024. 5

[51] J. Lawrence, J. Bernal, and C. Witzgall, “A purely algebraic justification of the kabsch-umeyama algorithm,” *J. Res. Natl. Inst. Stand. Technol.*, vol. 124, p. 1, 2019. 5

[52] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, “Rlbench: The robot learning benchmark & learning environment,” *IEEE RA-L*, vol. 5, no. 2, pp. 3019–3026, 2020. 5