IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

VERM: Leveraging Foundation Models to Create a Virtual Eye for Efficient 3D Robotic Manipulation

Yixiang Chen, Yan Huang, Keji He, Peiyan Li, and Liang Wang

***Abstract*—When performing 3D manipulation tasks, robots have to execute action planning based on perceptions from multiple fixed cameras. The multi-camera setup introduces substantial redundancy and irrelevant information, which increases computational costs and forces the model to spend extra training time extracting crucial task-relevant details. To filter out redundant information and accurately extract task-relevant features, we propose the VERM (Virtual Eye for Robotic Manipulation) method, leveraging the knowledge in foundation models to imagine a virtual task-adaptive view from the constructed 3D point cloud, which efficiently captures necessary information and mitigates occlusion. To facilitate 3D action planning and fine-grained manipulation, we further design a depth-aware module and a dynamic coarse-to-fine procedure. Extensive experimental results on both simulation benchmark RLBench and real-world evaluations demonstrate the effectiveness of our method, surpassing previous state-of-the-art methods while achieving 1.89× speedup in training time and 1.54× speedup in inference speed.**

***Index Terms*—Deep learning for visual perception, deep learning in grasping and manipulation, learning from demonstration.**

I. Introduction

3D ROBOTIC manipulation [1], [2], [3] has been widely studied due to its wide applications in industrial production and daily life. Unlike 2D pick-and-place tasks, 3D manipulation requires a comprehensive understanding of the 3D space using information from multiple fixed cameras. Recently, new progress [3], [4], [5] has been made for more accurate and efficient 3D robotic manipulation using diverse visual representations.

The specific views containing key information are crucial for robots performing 3D manipulation tasks. Robots can identify task-relevant details more clearly and mitigate potential occlusion through such views, similar to how humans choose where to interact with objects in daily life. However, in existing robotic manipulation environments, multiple fixed cameras often lead to substantial redundancy and irrelevant information in the input data, as they capture overlapping or unnecessary visual content.

Some researchers have recognized this issue and attempted to better merge information from different cameras. Some works [6], [7], [8], [9], [10] have projected RGB-D images from multiple cameras into a unified 3D space using either voxel or point cloud. Although these methods are able to reserve key information, the vast 3D representation still contains redundant background and irrelevant contents, making it difficult to identify the task-relevant information. Additionally, training the entire 3D representation is time-consuming, which costs about 16 days on 8 V100 GPUs [3]. To address these issues, other works [4], [5], [11] have alternatively projected the point cloud onto predefined virtual camera planes selected by human expertise, enabling re-rendering from views containing most useful information for specific tasks. However, the selection of these camera planes relies on expert knowledge, and they might not be effective for new tasks.

Different from these methods, humans can leverage their extensive knowledge to determine the task-adaptive view for manipulation in various tasks with just a glimpse and imagination. This topic has been extensively studied in the field of cognitive science for a long time [12], focusing on how our eyes automatically select where to look and the role of eye movements in visually guided behavior. Inspired by this, we propose the **VERM** (**V**irtual **E**ye for **R**obotic **M**anipulation) method, leveraging the knowledge in the large multimodal foundation model GPT-4o [13] to guide robots in selecting the task-adaptive view from a virtual camera pose, which captures the key information for manipulation and mitigates occlusion. Specifically, VERM first predicts a virtual camera plane based on the observations from multiple fixed real cameras and textual prompts using GPT-4o and then obtains the corresponding virtual camera image from the constructed 3D point cloud. Finally, this single image is used to guide the policy for generating actions. With this *predict, obtain, and guide* pipeline, the number of input tokens is

Received 16 May 2025; accepted 12 December 2025. Date of publication 12 January 2026; date of current version 20 January 2026. This article was recommended for publication by Associate Editor M. Saveriano and Editor A. Faust upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62322607, Grant 62236010, and Grant 62276261, in part by Beijing Natural Science Foundation under Grant L252033, in part by the Taishan Scholar Foundation of Shandong Province under Grant tsqn202507043, in part by the Natural Science Foundation of Shandong Province under Grant ZR2025QC1566, and in part by FiveAges Grant. *(Corresponding authors: Yan Huang; Liang Wang.)*

Yixiang Chen, Peiyan Li, and Liang Wang are with the New Laboratory of Pattern Recognition (NLPR), State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences, Beijing 100045, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100045, China (e-mail: yixiang.chen@cripac.ia.ac.cn; wangliang@nlpr.ia.ac.cn).

Yan Huang is with the New Laboratory of Pattern Recognition (NLPR), State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences, Beijing 100045, China, and with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100045, China, and also with the FiveAges, Beijing 100190, China (e-mail: yhuang@nlpr.ia.ac.cn).

Keji He is with Shandong University, Jinan 250100, China.

More results can be found on our project website: https://verm-ral.github.io/.

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2026.3652073, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3652073

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

significantly reduced, leading to shorter training time and faster inference speed without compromising performance.

Unlike other foundation-model-enhanced tasks such as task planning [14], [15], the proposed VERM overcomes two unique key challenges for 3D robotic manipulation. First, robots operate in 3D space, thus they heavily rely on depth information for spatial trajectory planning, so we design a depth-aware module to extend 2D action prediction from the single image into the third dimension. Second, robots require fine-grained manipulation to accurately complete given tasks. To address this, we propose a dynamic coarse-to-fine adjustment mechanism: when the model identifies a task-critical phase (e.g., precise alignment), it automatically triggers viewpoint zooming to focus on local regions of interest. This hierarchical refinement process allows iterative optimization of initial coarse actions only when necessary, improving manipulation success rates in high-precision tasks while maintaining efficiency.

Our contributions can be summarized as follows:

- We propose a structured prompting framework that reformulates GPT-4o into a spatial reasoning agent for viewpoint selection. By encoding environment context and instructions, the model outputs grounded camera poses that capture task-relevant details and reduce occlusion, revealing the 3D spatial reasoning capability of large multimodal models.
- We validate the generality of our approach across different foundation models, including GPT-4o, Qwen2.5, and Claude 3.5, showing that VERM can operate as a plug-and-play spatial reasoning module without fine-tuning or architectural modification.
- We design a depth-aware module to extend action planning to 3D space, and a dynamic coarse-to-fine procedure to refine actions, facilitating 3D and fine-grained manipulation.
- The proposed VERM method demonstrates effectiveness over previous state-of-the-art methods on both simulation benchmark RLBench [16] and real-world evaluations, achieving **1.89**× speedup in training time and **1.54**× speedup in inference speed.

II. Related Work

Foundation Models for Robotics: Foundation models are increasingly adopted in robotics [17], [18], offering prior knowledge that supports perception [19], [20], [21], [22], code generation [23], [24], [25], [26], [27], [28], [29], and task planning [1], [2], [14], [15], [30], [31]. SayCan [30] decomposes human instructions into sub-tasks grounded in the physical world, while Code-as-Policies [23] generates executable policies directly from LLMs. VoxPoser [26] combines LLMs with SAM [20] for zero-shot generalization, and foundation models like GPT-4 V have enabled integrated vision-language planning [14], [15]. Unlike prior works focused on high-level reasoning or skill planning, we use GPT-4o [13] to tackle viewpoint selection: generating a virtual camera pose that integrates multi-view observations and minimizes occlusion. We design a structured prompting pipeline to effectively query GPT-4o's spatial reasoning capabilities for this purpose.

Vision-based Manipulation in 3D Space: The selection of an appropriate visual scene representation is crucial for vision-based manipulation tasks in the 3D space. CLIPort [32] has used RGB-D images as input and predicted the 2D affordance map and an estimated depth value. In contrast, other works [3], [6], [8], [9] have alternatively employed 3D representations. Among them, C2F-ARM [6] and Peract [3] have utilized voxel maps, while Act3D [8] and 3D Diffuser Actor [9] have used 3D point cloud as their visual representations. These methods, though effective, can be computationally expensive. Alternatively, RVT series [4], [5] and VIHE [11] have projected RGB-D images captured from multiple cameras into a unified point cloud, which is then re-projected onto several predefined image planes. However, the selection of image planes requires human intervention and lacks flexibility for different tasks. Our work is closely related to RVT [4] and RVT-2 [5], yet we try to identify the task-adaptive view that contains all necessary information for task completion. This innovation reduces human intervention and simplifies the input to a single 2D image, thus greatly accelerating training and inference.

High-Precision Manipulation: It is challenging to perform tasks requiring high precision. For instance, inserting pegs demands precision without any deviation. Previous research has explored various methods to achieve this. C2F-ARM [6] has employed a coarse-to-fine approach, initially using a low-resolution voxel map to locate a coarse position, then refining the resolution around this position to enhance precision. RVT-2 [5] has also utilized the coarse-to-fine approach but with images as input. Building upon these foundations, our framework introduces two key innovations: (1) a single-image input architecture that reduces computational complexity and accelerates both training and inference, and (2) a *dynamic* coarse-to-fine mechanism that selectively triggers fine-grained refinement only during task-critical phases. Unlike prior methods that apply refinement uniformly, our approach adapts to the task context, improving efficiency without sacrificing precision.

III. VERM

A. Overview

Our objective is to perform a language-conditioned imitation learning task utilizing a dataset $D = \{\zeta_1, \zeta_2, \ldots, \zeta_n\}$ of n expert demonstrations. Each demonstration ζ_i consists of sequences of RGB-D observations $\{o^i_{1,\ldots,m_i}\}$ captured from fixed scene cameras, expert action sequences $\{a^i_{1,\ldots,m_i}\}$, and a language description l_i, with episode length m_i. Our approach involves training a policy that, given natural language instructions and RGB-D observations, predicts an 8-dimensional action vector of the end-effector. This vector includes a 3-DOF position, 3-DOF rotation, an 1-DOF gripper state (open or closed), and an 1-DOF collision parameter (indicating whether the motion planner considers potential collisions with scene objects). The motion planner calculates the path between consecutive predicted actions.

Our method can be summarized as follows. First, we develop a prompt-based paradigm to find the task-adaptive camera pose for completing the task using GPT-4o, as illustrated in Fig. 1.

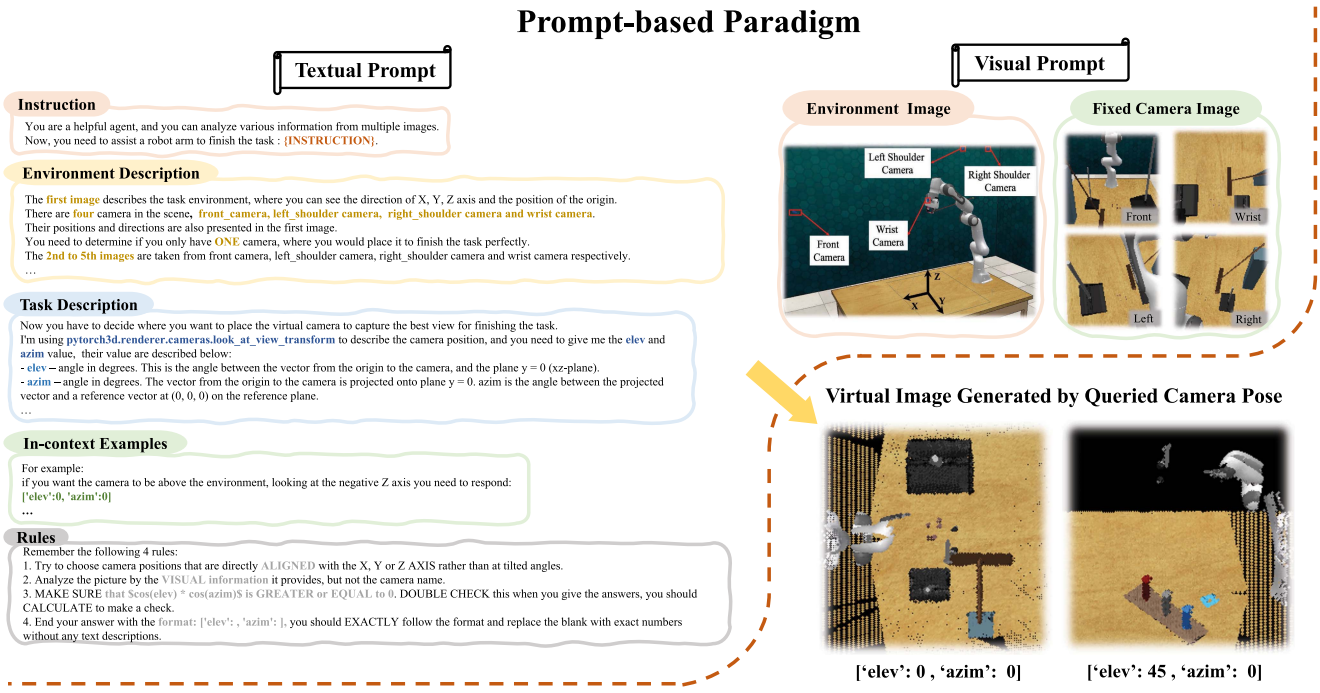

Fig. 1. The prompt-based paradigm for querying virtual camera poses using GPT-4o.

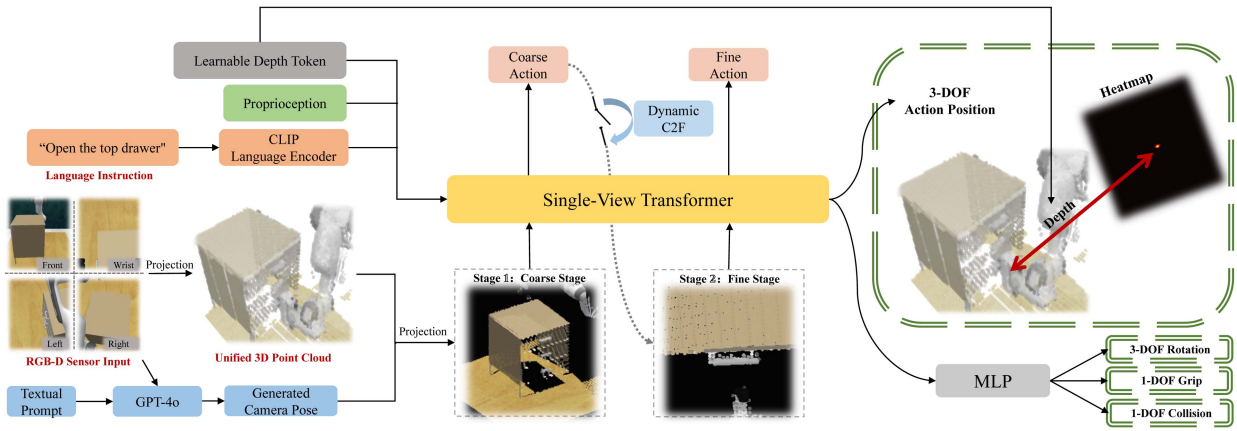

Fig. 2. Policy network of the proposed VERM.

Next, given the two-dimensional nature and limited detail of the single image from the first step, we further incorporate a depth-aware module for 3D manipulation and a coarse-to-fine procedure for precise action planning, as illustrated in Fig. 2.

B. Camera Pose Selection

It is non-trivial to prompt GPT-4o for querying camera poses, as it requires a comprehensive understanding of the 3D space. We develop a prompt-based paradigm for camera pose selection, which is depicted in Fig. 1. The left panel organizes a textual prompt into distinct sections, while the top-right panel presents two visual prompts, where the image on the left outlines the workspace with original fixed camera locations and axes (using the SoM (Set-of-Mark) [33] prompting technique), while the right one displays original RGB images from these cameras. GPT-4o processes these prompts for camera pose selection.

We carefully structure the textual prompt into four distinct parts: environment description, task description, in-context examples, and rules.

Environment Description: This section provides a basic overview of the fixed camera poses within the scene, describes the position of origin, and defines the orientations of axes in the visual prompt. Initial attempts involved specifying camera intrinsics and extrinsics, but we find that GPT-4o struggles with spatial relationships using these parameters. Instead, a visual representation of the environment using SoM could be easier for the model to interpret.

Task Description: In this section, we define camera pose using two parameters: $elev$ and $azim$. The parameter $elev$ describes

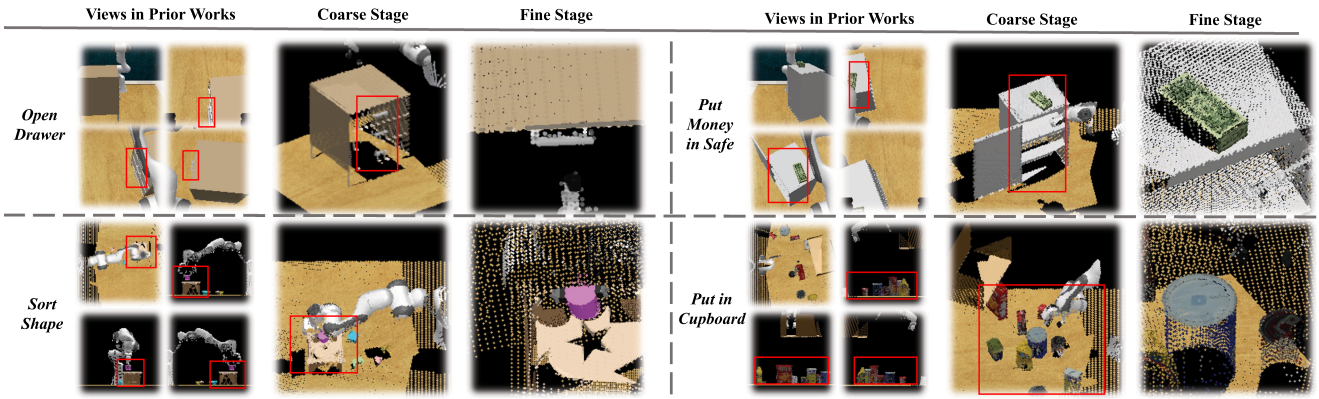

Fig. 3. Visualization of action prediction of VERM in RLBench.

the angle between the vector from the origin to the camera and the $xz-plane$, while $azim$ specifies the angle between the projected vector and a reference vector at $(0,0,0)$ on the reference plane. These angles help define the camera's direction, pointing towards the origin from a fixed distance.

In-context Examples: We include in-context examples in the prompt to help the model better understand the spatial relationships in the environment. These examples are drawn from three cameras defined in the RVT-2 [4] setup.

Rules: We establish four rules to refine the output from GPT-4o to eliminate undesirable results. Firstly, we prefer camera alignments along axes to facilitate easier depth information capture. Secondly, we have observed that the model sometimes shortcuts its analysis by referring to camera labels (e.g., front_camera) rather than the provided visual information; hence, we emphasize the importance of analyzing visual information. The third and fourth rules ensure that the camera are positioned to look downwards from above the table and constrain the output format.

To get the ideal camera pose from GPT-4o, we combine the textual prompt with five visual prompts—one for the environment description and four from original RGB-D inputs. Examples of virtual images generated by the queried camera poses are shown in the bottom-right corner of Fig. 1. Although these images capture key task-relevant information, they are insufficient for 3D manipulation due to two factors: the 2D input lacks the necessary depth information for 3D action planning, and the coarse resolution cannot support fine-grained tasks like the peg insertion shown on the right. We address these limitations by incorporating a depth-aware module and a dynamic coarse-to-fine procedure in the policy network, as detailed in the following section.

C. Policy Network

The architecture of the policy network is depicted in Fig. 2. The original RGB-D inputs are transformed into a unified 3D point cloud, which is then projected onto a virtual camera plane as specified by GPT-4o. We use this virtual image, combined with language instructions and proprioceptive (robot state and time information) inputs, to predict a coarse action. This prediction involves generating a 2D heatmap on the virtual image and estimating a corresponding depth value. Subsequently, we enable the camera to zoom in on the coarse action, allowing it to focus on the critical local content and refine the initial coarse action for higher precision.

Dynamic Coarse-to-Fine Module: We first transform the original RGB-D images $o_{front}, o_{l_shoulder}, o_{r_shoulder}, o_{wrist}$ into a unified 3D point cloud, which is then re-projected onto a virtual camera plane specified by GPT-4o's $elev$ and $azim$ parameters to generate a single global image o_{global}. This image merges all views to optimally capture task-relevant details. While o_{global} is sufficient for most tasks, high-precision actions benefit from finer views. In such cases, we apply a zoom-in operation by centering the point cloud on the coarse prediction and scaling it, without changing camera orientation.

However, not all stages of a task demand the same level of precision. For instance, in the *stack blocks* task, free-space motions such as transporting the block can be executed with coarse predictions, whereas fine-grained accuracy is essential during grasping or placement. To handle this, we introduce a dynamic coarse-to-fine (C2F) inference module that selectively applies refinement only when needed.

Unlike prior C2F approaches that apply refinement at every step, our method uses the discrepancy between coarse and fine predictions during training to identify task-critical phases. If the predicted translation or rotation differs beyond a threshold (0.01 m or 5°), the sample is labeled as requiring refinement. A lightweight predictor is trained to classify such cases and is used at inference time to decide whether to activate the fine stage. This selective refinement strategy ensures that computational resources are focused on high-precision moments while avoiding unnecessary overhead during simpler motions, improving both accuracy and efficiency.

Depth-Aware Module: In this module, we incorporate learnable depth tokens to predict depth value for action planning in 3D space. The 3-DOF action position is defined on the right side of Fig. 2. We predict the heatmap in the same space as the observation space and estimate a depth value using depth tokens. We discretize the 3-DOF rotation into 5-degree bins for each axis to predict the rotation.

Action prediction is performed using Transformer [34], which processes learnable depth tokens, language tokens, and image tokens together through an attention mechanism.

Keypoint Selection: Following prior work [3], [6], we reduce each trajectory to a sparse set of keypoints to avoid training on noisy intermediate steps. Keypoints are defined by two heuristics: (1) changes in end-effector state (e.g., grasp or release), and (2) near-zero velocity, typically before or after critical transitions.

Training: The ground truth for the heatmaps is generated from a Gaussian distribution centered on the expert action. The rotation, gripper state, and collision parameters are trained using cross-entropy loss with one-hot encoded expert actions. We also formulate depth prediction as a classification task to align it with other loss metrics. The loss is defined in Equation (1), where $\mathcal{L}_i(i \in \{trans, rot, open, collision, depth\})$ refers to the translation, rotation, gripper open, collision and depth loss respectively and $\mathcal{L}_{dyn\_inf}$ refers to the loss of the dynamic coarse-to-fine indicator.

$$\mathcal{L} = \mathcal{L}_{trans} + \mathcal{L}_{rot} + \mathcal{L}_{open} + \mathcal{L}_{collision} + \mathcal{L}_{depth} + \mathcal{L}_{dyn\_inf} \quad (1)$$

IV. Experiment Results

We conduct our experiments using the simulation benchmark RLBench (Sec IV-A), as well as real-world evaluations (Sec IV-B).

A. Rlbench

Simulation Setup: Our experimental setup on RLBench [16] is designed in line with PerAct [3], which uses CoppeliaSim [36] to simulate a variety of tasks from RLBench. We employ a Franka Panda robot equipped with a parallel gripper to perform tasks introduced by PerAct, which include diverse activities such as picking and placing, tool manipulation, drawer opening, and high-accuracy peg insertions. For each task, variations are created based on associated language descriptions to enhance task complexity. Visual observations are captured from four noiseless RGB-D cameras placed at strategic locations (front, left shoulder, right shoulder, and wrist), each providing 128×128 resolution images. The path between predicted action and present action is calculated using a sampling-based motion planner [37], as used in [3], [4], [6], [8], [11], which facilitates the generation of joint space actions needed to complete the tasks effectively.

Compared Methods: We compare our method against eight state-of-the-art methods and all methods use the same RGB-D inputs from four fixed cameras for fair comparison:

1) C2F-ARM-BC [6], which has converted RGB-D images into multi-resolution voxel representations and predicted key-frame actions through a coarse-to-fine strategy;
2) PerAct [3], which has voxelized RGB-D images and employed a Perceiver transformer for action prediction;
3) HiveFormer [35], which has directly used images captured from original cameras;
4) PolarNet [7], which has used dense point representation for action prediction;
5) RVT [4], which has generated five global orthographic views by human expertise and utilizes a multi-view transformer for predicting actions;

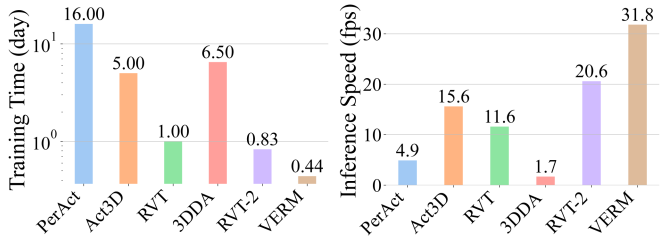

Fig. 4. *Left:* Training time (day) in log scale. *Right:* Inference speed (fps).

6) Act3D [8], which has leveraged pre-trained image features and applied relative cross-attention mechanisms on point cloud for action detection;
7) 3D Diffuser Actor (3DDA) [9], which has used diffusion policy to predict actions on 3D point cloud;
8) RVT-2 [5], which has been built on RVT [4] with more efficient implementation and coarse-to-fine strategy.

Implementation Details: We use the ResNet50 variant of CLIP [19] to encode language instructions. Images are rendered from point clouds via PyTorch3D orthographic cameras [4] at 224×224 resolution. The Transformer takes 256 image tokens (patch size 14×14), 77 language tokens, and 36 learnable depth tokens, with 8 layers. Camera view selection is performed once per task using the OpenAI `gpt-4o` API.

Training and Evaluation Details: Each RLBench task contains 100 expert demonstrations, collected using four fixed RGB-D cameras per scene. Each trajectory consists of 50–150 timesteps, from which we extract 2–12 keypoints. Training is conducted on 8 NVIDIA V100 (16 GB) GPUs using the LAMB optimizer [38], with batch size 640 (80×8) and learning rate 2.4×10^{-3}. Evaluation follows PerAct [32], using 25 trajectories per task. A trial is marked successful if the robot meets predefined conditions; failure occurs if the planner yields infeasible actions or exceeds time limits. To account for randomness in sampling-based planning, we repeat each trial five times and report both mean and variance of success rates.

Results: The training time and inference speed of VERM, along with those of previous works,[1] are presented in Fig. 4. VERM achieves **1.89**× speedup in training time and **1.54**× speedup in inference speed than the previous state-of-the-art method RVT-2 using the same GPU setup, which demonstrates that single-image input not only enables efficient learning but also opens up new possibilities for real-time control.

In addition to the reduced training time and faster inference speed, VERM does not compromise task success rate, as shown in Table I. VERM surpasses RVT-2 by 1.4% in average task success rate and performs best in 11 out of 17 tasks. This indicates that the single image, informed by GPT-4o, is sufficient for acquiring all task-relevant content. VERM alleviates the need to pre-define multiple virtual camera planes in the RVT series method (Note that the 3DDA method adopts an original image size of 256, whereas VERM and the RVT series use 128).

[1] PerAct, RVT, RVT-2, and VERM use 8 16 GB V100 GPUs, while Act3D uses 8 32 GB V100 GPUs and 3DDA uses 7 40 GB A100 GPUs. Among these methods, VERM aligns with the least demanding GPU setup.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
MULTI-TASK PERFORMANCE ON RLBENCH

	Avg. Success ↑	Stack Blocks	Open Drawer	Slide Block	Sweep to Dustpan	Meat off Grill	Turn Tap	Put in Drawer	Close Jar
C2F-ARM-BC [6]	16.9	0	20.0	16.0	0.0	20.0	68.0	4.0	24.0
PerAct [3]	52.3	$26.4_{\pm 3.2}$	$88.0_{\pm 5.7}$	$74.0_{\pm 13.0}$	$52.0_{\pm 0.0}$	$70.4_{\pm 2.0}$	$88.0_{\pm 4.4}$	$51.2_{\pm 4.7}$	$55.2_{\pm 4.7}$
HiveFormer [35]	48.0	8.0	52.0	64.0	28.0	**100.0**	80.0	68.0	52.0
PolarNet [7]	48.7	4.0	84.0	56.0	52.0	**100.0**	80.0	32.0	36.0
RVT [4]	65.0	$28.8_{\pm 3.9}$	$71.2_{\pm 6.9}$	$81.6_{\pm 5.4}$	$72.0_{\pm 0.0}$	$88.0_{\pm 2.5}$	$93.6_{\pm 4.1}$	$88.0_{\pm 5.7}$	$52.0_{\pm 2.5}$
Act3D [8]	68.4	12.0	93.0	93.0	92.0	94.0	94.0	90.0	93.0
3DDA [9]	83.5	$68.3_{\pm 3.3}$	**89.6**$_{\pm 4.1}$	$97.6_{\pm 3.2}$	$84.0_{\pm 4.4}$	$96.8_{\pm 1.6}$	**99.2**$_{\pm 1.6}$	$96.0_{\pm 3.6}$	$96.0_{\pm 2.5}$
RVT-2 [5]	82.2	$80.0_{\pm 2.86}$	$74.0_{\pm 11.8}$	$92.0_{\pm 2.8}$	**100.0**$_{\pm 0.0}$	$99.0_{\pm 1.7}$	$99.0_{\pm 1.7}$	$96.0_{\pm 0.0}$	**100.0**$_{\pm 0.0}$
VERM(ours)	**83.6**	**80.8**$_{\pm 4.4}$	**89.6**$_{\pm 3.8}$	**99.2**$_{\pm 2.6}$	$95.2_{\pm 1.8}$	**100.0**$_{\pm 0.0}$	$98.4_{\pm 2.2}$	**100.0**$_{\pm 0.0}$	$96.8_{\pm 3.3}$

	Screw Bulb	Put in Safe	Place Wine	Put in Cupboard	Sort Shape	Push Buttons	Insert Peg	Place Cups	Drag Stick
C2F-ARM-BC [6]	8.0	12.0	8.0	0.0	8.0	72.0	4.0	0.0	24.0
PerAct [3]	$17.6_{\pm 2.0}$	$86.0_{\pm 3.6}$	$44.8_{\pm 7.8}$	$28.0_{\pm 4.4}$	$16.8_{\pm 4.7}$	$92.8_{\pm 3.0}$	$5.6_{\pm 4.1}$	$2.4_{\pm 2.2}$	$89.6_{\pm 4.1}$
HiveFormer [35]	8.0	76.0	80.0	32.0	8.0	84.0	0.0	0.0	76.0
PolarNet [7]	44.0	84.0	40.0	12.0	12.0	96.0	4.0	0.0	92.0
RVT [4]	$48.0_{\pm 5.7}$	$91.2_{\pm 3.0}$	$91.0_{\pm 5.2}$	$49.6_{\pm 3.2}$	$36.0_{\pm 2.5}$	$100.0_{\pm 0.0}$	$11.2_{\pm 3.0}$	$4.0_{\pm 2.5}$	$99.2_{\pm 1.6}$
Act3D [8]	47.0	95.0	80.0	51.0	8.0	99.0	27.0	3.0	92.0
3DDA [9]	$82.4_{\pm 2.0}$	$97.6_{\pm 2.0}$	$93.6_{\pm 4.8}$	**85.6**$_{\pm 4.1}$	$44.0_{\pm 4.4}$	$98.4_{\pm 2.0}$	**65.6**$_{\pm 4.1}$	$24.0_{\pm 7.6}$	**100.0**$_{\pm 0.0}$
RVT-2 [5]	$88.0_{\pm 4.9}$	$96.0_{\pm 2.8}$	$95.0_{\pm 3.3}$	$66.0_{\pm 4.5}$	$35.0_{\pm 7.1}$	**100.0**$_{\pm 0.0}$	$40.0_{\pm 0.0}$	$38.0_{\pm 4.5}$	$99.0_{\pm 1.7}$
VERM(ours)	**93.6**$_{\pm 3.6}$	**100.0**$_{\pm 0.0}$	**96.0**$_{\pm 2.8}$	$55.2_{\pm 6.6}$	**47.2**$_{\pm 5.9}$	**100.0**$_{\pm 0.0}$	$30.4_{\pm 4.6}$	**40.0**$_{\pm 2.8}$	$99.2_{\pm 1.8}$

TABLE II
ABLATION STUDY

#	Core Design					Rendering Parameters				Avg. Succ.
	Cam. Pose				C2F	Img. Size		Zoom	Align	
	GPT-4o	Front	Top	Right		112	224	In	Axis	
1	✓				✓		✓	✓	✓	**83.6**
2		✓			✓		✓	✓	✓	58.3
3			✓		✓		✓	✓	✓	70.9
4				✓	✓		✓	✓	✓	55.1
5	✓						✓	✓	✓	56.7
6	✓				✓	✓		✓	✓	71.3
7	✓				✓	✓	✓		✓	59.2
8	✓				✓		✓	✓		66.4

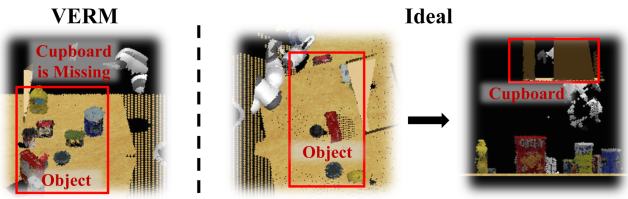

Fig. 5. Example failure cases.

TABLE III
EVALUATION OF CAMERA POSE GENERATION CAPABILITIES OF DIFFERENT FOUNDATION MODELS

Model	Deviation from GPT-4o	Success Rate
GPT-4o [13]	0°	**83.6%**
Qwen2.5 [39]	11°	80.3%
Claude 3.5 Sonnet [40]	8°	81.2%

Qualitative Analysis: Fig. 3 shows that VERM-generated views effectively integrate information from multiple fixed cameras while avoiding occlusion. In *open drawer* and *put money*, the selected view reveals the full object (e.g., entire handle) that is partially missing in original views. In *sort shape*, a slight rotation exposes occluded holes. These results demonstrate VERM's ability to generate concise, task-relevant views that support accurate action prediction.

Ablation Study: The ablation results in Table II show that model #1 represents the full VERM design. Models #2–#4 test single-camera inputs from the RVT-2 setup, where the GPT-4o–generated global view outperforms all predefined cameras by adapting to task variations. The top camera performs best among the fixed ones but remains limited under occlusion or multi-object orientations. Removing the coarse-to-fine process (#5) reduces precision, while the rendering settings in #6 and #7 confirm that low resolution and missing zoom-in both degrade performance. Model #8 removes the axis-alignment constraint, leading to a performance drop, indicating that alignment improves depth prediction while still allowing necessary rotations to reduce occlusion.

Cross-Model Generalization: To evaluate the generalizability of VERM across different foundation models, we additionally tested our method using Qwen2.5 (open-source) [39] and Claude 3.5 Sonnet [40]. Since these models tend to generate suboptimal poses on the first attempt, we apply an iterative refinement strategy with self-verification feedback to adjust camera parameters. As shown in Table III, both models achieve comparable task success rates (80.3% and 81.2%, respectively) to GPT-4o (83.6%), despite moderate angular deviations. These results

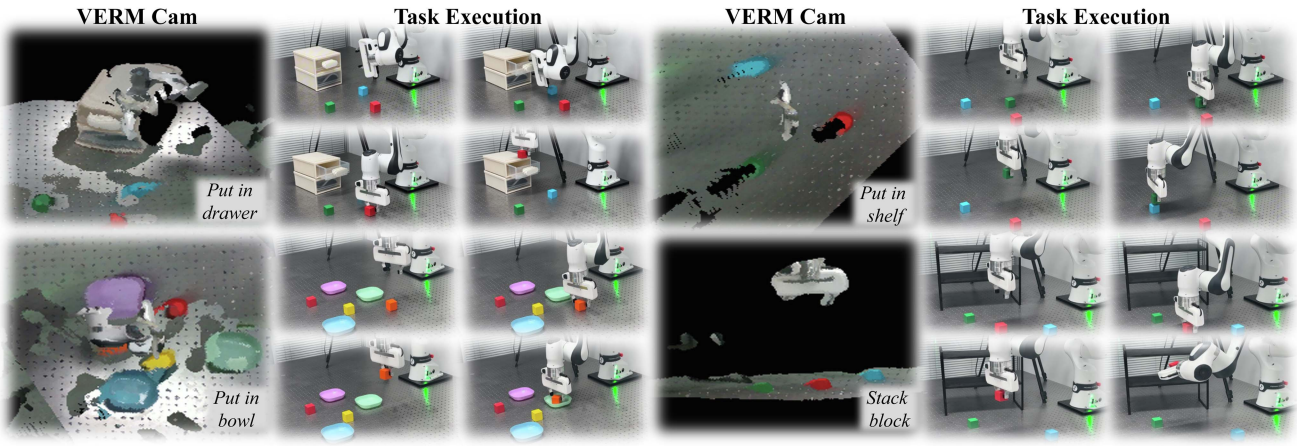

Fig. 6. Visualization of action prediction of VERM in real-world.

confirm that VERM is compatible with alternative foundation models while maintaining robust performance.

Failure Cases and Model Hallucinations: Fig. 5 shows typical failure cases. In tasks like *put in cupboard*, the object and target container are visible from different angles, making it difficult to capture all relevant details in a single view. As our method queries the foundation model only once at the task's start, it may miss critical cues needed in later stages. We address this with a dynamic re-querying strategy that updates the viewpoint mid-execution, raising success of *put in cupboard* task from 55.2% to 66.4%, although with higher computational cost.

We also observe hallucinations from GPT-4o, such as infeasible viewpoints (e.g., under the table) or overreliance on camera labels instead of visual cues. To reduce this, we enforce prompt constraints: (i) reasoning must be based on visual input only, and (ii) generated poses must satisfy geometric validity ($z > 0$). A self-verification loop further filters out unsatisfactory views by checking the rendered result against criteria. These measures improve robustness and support generalization to other models like Qwen and Claude, which tend to hallucinate more frequently.

B. Real-World Evaluation

Real-world Setup: We evaluate our VERM methods by comparing them with RVT [4] and RVT-2 [5] on a real-world manipulation setup, similar to those used in previous studies. This setup includes a statically mounted Franka Research 3 arm and 2 stationary Intel Realsense D435i RGB-D cameras for third-person viewing. We calibrate the robot camera extrinsic and convert the two perceived point clouds into the unified robot's base frame before feeding them into the VERM network. For a given target gripper action, we utilize Deoxys [41] to guide the robot to the target through trajectory generation and feedback control.

Task Definition and Data Collection: We evaluate VERM on eight real-world manipulation tasks: *stack blocks*, *press sanitizer*, *place block in bowl*, *place object in drawer*, *place object in shelf*, *flip cup*, *close drawer*, and *open cabinet*. Each task includes 1–7 object variants defined by language instructions. Each task contains 100 demonstrations, augmented via 3D point cloud transformations to simulate diverse spatial layouts. Evaluation is conducted over 10 trials per configuration using unseen object placements.

TABLE IV
MULTI-TASK PERFORMANCE ON REAL-WORLD EVALUATION

Task	# of vari.	Models			
		RVT (15 Traj)	RVT-2 (15 Traj)	VERM (15 Traj)	VERM (100 Traj)
Stack blocks	3	70%	**80%**	**80%**	**80%**
Press sanitizer	1	70%	80%	**90%**	80%
Put block in bowl	5	40%	60%	70%	**80%**
Put object in drawer	7	30%	50%	**70%**	**70%**
Put object in shelf	7	50%	70%	**80%**	**80%**
Flip cup	1	20%	60%	**80%**	70%
Close drawer	1	50%	70%	90%	**100%**
Open cabinet	1	40%	70%	70%	**80%**
All tasks	26	46.25%	67.50%	78.75%	**80.00%**

Results: Table IV reports the success rates under two settings: using 15 and 100 demonstrations per task. VERM achieves strong performance with just 15 trajectories, already outperforming RVT and RVT-2 in most tasks while significantly reducing both training time and inference latency. When trained with 100 demonstrations, performance further improves slightly, suggesting that our method is highly data-efficient and generalizes well with limited supervision.

Fig. 6 shows action predictions in the real world. Despite using low-cost RGB-D sensors and encountering lighting or point cloud noise, the virtual views generated by VERM still preserve key task-relevant details, enabling reliable execution.

V. CONCLUSION

We present VERM, a framework that leverages GPT-4o to generate a virtual eye for robotic manipulation. By fusing multi-camera inputs into a task-adaptive view, VERM reduces perception to a single image while maintaining high success

rates. A dynamic coarse-to-fine module further refines actions, enabling efficient and accurate manipulation.

Limitations and Future Work: Our approach assumes fixed multi-camera setups and queries the viewpoint only at the beginning. Future work may explore dynamic view selection over time for better adaptability, as well as incorporating history observations to support long-horizon planning. We also plan to evaluate VERM on contact-rich manipulation tasks using datasets like REASSEMBLE [42], to further assess generalization under physical interaction constraints.

REFERENCES

[1] A. Brohan et al., "RT-1: Robotics transformer for real-world control at scale,"in *Proc. Robotics., Sci. Syst.*, 2023.

[2] B. Zitkovich et al., "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. Conf. Robot Learn.*, 2023, pp. 2165–2183.

[3] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-Actor: A multi-task transformer for robotic manipulation," in *Proc. Conf. Robot Learn.*, 2023, pp. 785–799.

[4] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox, "RVT: Robotic view transformer for 3D object manipulation," in *Proc. Conf. Robot Learn.* 2023, pp. 694–710.

[5] A. Goyal, V. Blukis, J. Xu, Y. Guo, Y.-W. Chao, and D. Fox, "RVT2: Learning precise manipulation from few demonstrations," in *Proc. Robot., Sci. Syst.*, 2024.

[6] S. James, K. Wada, T. Laidlow, and A. J. Davison, "Coarse-to-fine Q-attention: Efficient learning for visual robotic manipulation via discretisation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 13739–13748.

[7] S. Chen, R. G. Pinel, C. Schmid, and I. Laptev, "PolarNet: 3D point clouds for language-guided robotic manipulation," in *Proc. Conf. Robot Learn.*, 2023, pp. 1761–1781.

[8] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3D: 3D feature field transformers for multi-task robotic manipulation," in *Proc. 7th Annu. Conf. Robot Learn.*, 2023, pp. 3949–3965.

[9] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3D diffuser actor: Policy diffusion with 3D scene representations," in *Proc. 8th Annu. Conf. Robot Learn.*, 2024, pp. 1949–1974.

[10] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations," in *Proc. Robotics, Sci. Syst.*, 2024.

[11] W. Wang, Y. Lei, S. Jin, G. D. Hager, and L. Zhang, "VIHE: Virtual in-hand eye transformer for 3D robotic manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2024, pp. 403–410.

[12] M. Hayhoe and D. Ballard, "Eye movements in natural behavior," *Trends Cogn. Sci.*, vol. 9, no. 4, pp. 188–194, 2005.

[13] A. Hurst et al., "GPT-4o system card," 2024, *arXiv:2410.21276*.

[14] N. Wake, A. Kanehira, K. Sasabuchi, J. Takamatsu, and K. Ikeuchi, "GPT-4V(ision) for robotics: Multimodal task planning from human demonstration," *IEEE Robot. Automat. Lett.*, vol. 9, no. 11, pp. 10567–10574, Nov. 2024.

[15] Y. Hu, F. Lin, T. Zhang, L. Yi, and Y. Gao, "Look before you leap: Unveiling the power of GPT-4v in robotic vision-language planning," 2023, *arXiv:2311.17842*.

[16] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "RLBench: The robot learning benchmark & learning environment," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 3019–3026, Apr. 2020.

[17] R. Firoozi et al., "Foundation models in robotics: Applications, challenges, and the future," *Int. J. Robot. Res.*, vol. 44, pp. 701–739, 2024.

[18] X. Xiao et al., "Robot learning in the era of foundation models: A survey," *Neurocomputing*, vol. 638, 2025, Art. no. 129963.

[19] A. Radford et al., "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8748–8763.

[20] A. Kirillov et al., "Segment anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.

[21] S. Liu et al., "Grounding DINO: Marrying dino with grounded pre-training for open-set object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 38–55.

[22] M. Minderer et al., "Simple open-vocabulary object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 728–755.

[23] J. Liang et al., "Code as policies: Language model programs for embodied control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 9493–9500.

[24] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "ChatGPT for robotics: Design principles and model abilities," *IEEE Access*, vol. 12, pp. 55682–55696, 2024.

[25] I. Singh et al., "ProgPrompt: Generating situated robot task plans using large language models," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 11523–11530.

[26] 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," in *Proc. Conf. Robot Learn.*, 2023, pp. 540–562.

[27] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, "ReKep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation," in *Proc. 8th Annu. Conf. Robot Learn.*, 2024, pp. 4573–4602.

[28] Y. Du et al., brian ichter, "Video language planning," in *Proc. 12th Int. Conf. Learn. Representations*, 2024.

[29] W. Zhao, J. Chen, Z. Meng, D. Mao, R. Song, and W. Zhang, "VLMPC: Vision-language model predictive control for robotic manipulation," in *Proc. Robotics, Sci. Syst.*, 2024.

[30] A. Brohan et al., "Do as i can, not as i say: Grounding language in robotic affordances," in *Proc. Conf. Robot Learn.*, 2023, pp. 287–318.

[31] D. Driess et al., "PaLM-E: An embodied multimodal language model," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 8469–8488.

[32] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and where pathways for robotic manipulation," in *Proc. Conf. Robot Learn.*, 2022, pp. 894–906.

[33] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, and J. Gao, "Set-of-mark prompting unleashes extraordinary visual grounding in GPT-4v," 2023, *arXiv:2310.11441*.

[34] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30.

[35] P.-L. Guhur, S. Chen, R. G. Pinel, M. Tapaswi, I. Laptev, and C. Schmid, "Instruction-Driven history-aware policies for robotic manipulations," in *Proc. Conf. Robot Learn.*, 2023, pp. 175–187.

[36] E. Rohmer, S. P. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1321–1326.

[37] S. Karaman and E. Frazzoli, "Sampling-Based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[38] Y. You et al., "Large batch optimization for deep learning: Training bert in 76 minutes," in *Proc. Int. Conf. Learn. Representations*, 2020.

[39] A. Yang et al., "Qwen2. 5 technical report," 2024, *arXiv:2412.15115*.

[40] Anthropic, "Claude 3.5 sonnet model card addendum," 2024. [Online]. Available: https://www-cdn.anthropic.com/fed9cc193a14b84131812372d8d5857f8f304c52/Model_Card_Claude_3_Addendum.pdf

[41] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "VIOLA: Object-centric imitation learning for vision-based robot manipulation," in *Proc. 6th Annu. Conf. Robot Learn.*, 2022.

[42] D. Sliwowski, S. Jadav, S. Stanovcic, J. Orbik, J. Heidersberger, and D. Lee, "Demonstrating REASSEMBLE: A multimodal dataset for contact-rich robotic assembly and disassembly," in *Proc. Robot., Sci. Syst.*, Los Angeles, CA, USA, Jun. 2025.