2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

VIHE: Virtual In-Hand Eye Transformer for 3D Robotic Manipulation

Weiyao Wang[1,†], Yutian Lei[2], Shiyu Jin[2], Gregory D. Hager[1] and Liangjun Zhang[2]

***Abstract*— In this work, we introduce the Virtual In-Hand Eye Transformer (VIHE), a novel method designed to enhance 3D manipulation capabilities through action-aware view rendering. VIHE autoregressively refines actions in multiple stages by conditioning on rendered views posed from action predictions in the earlier stages. These virtual in-hand views provide a strong inductive bias for effectively recognizing the correct pose for the hand, especially for challenging high-precision tasks such as peg insertion. On 18 manipulation tasks in RLBench simulated environments, VIHE achieves a new state-of-the-art, with a 12% absolute improvement, increasing from 65% to 77% over the existing state-of-the-art model using 100 demonstrations per task. In real-world scenarios, VIHE can learn manipulation tasks with just a handful of demonstrations, highlighting its practical utility. Videos and code implementation can be found at our project site: https://vihe-3d.github.io.**

I. Introduction

Achieving mastery in manipulating objects in 3D environments is a foundational goal in the quest for intelligent real-world robotic systems. While machine learning and computer vision have propelled significant advancements in robotic manipulation [1], [2], [3], [4], [5], the challenge of crafting an effective observation space for effective learning in 3D persists. Current methodologies in the realm of robotic manipulation employ various types of 3D representations. Recently in the space of end-to-end vision-based imitation learning in 3D, PerAct [6] utilizes voxel-based representations, which, although powerful, suffer from computational inefficiencies due to the cubic scaling of voxels. RVT [7] employs multi-view orthographic images but faces difficulties in tasks that demand high-precision 3D reasoning. Act3D [8] leverages point clouds for 3D representation but also computationally suffers from large number of sampled points and neglects the potential advantages of spatial biases in manipulation tasks.

We observe that existing approaches often treat the 3D workspace uniformly, neglecting the naturally occurring inductive bias that the space near the end-effector holds significant importance for manipulation tasks. Previous research has underscored the value of an in-hand perspective: for instance, [9] demonstrates that an in-hand view reveals more task-relevant details, which is particularly advantageous for high-precision tasks. Similarly, [10] shows that incorporating

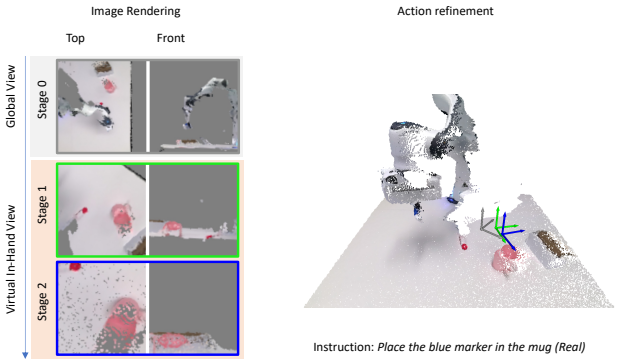

Fig. 1. Visual example of VIHE in real-world. Our method iteratively refines its 3D action prediction (right) by rendering 2D in-hand views based on the previous stage predictions (left). Color coding of gray, green, and blue represent three action prediction stages respectively.

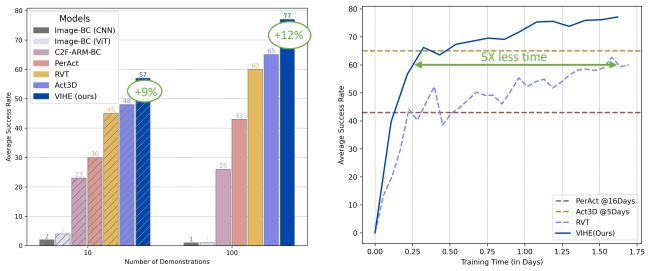

Fig. 2. VIHE scales and performs better than RVT, PerAct, and Act3D among other baselines. Attribute to the inductive bias from in-hand views, VIHE also require 5X less time to achieve on-par performance to the previous SOTA method.

an in-hand view can mitigate distractions irrelevant to gripper actions, thereby improving generalization. However, these studies are primarily limited to 2D image-based manipulation [9], [10], utilizing a physical camera attached to the robot's end-effector. Consequently, we aim to extend the inductive bias of the end-effector location to better structure observations for 3D manipulation tasks.

We introduce the Virtual In-Hand Eye Transformer (VIHE) which utilizes an iterative process to refine action predictions, leveraging rendered in-hand views at each stage. In each refinement stage, we render local images based on the predicted action in the previous stage. Conditioned on these in-hand perspectives, the model subsequently predicts relative SE(3) transformations to refine the end-effector pose from the previous stage. The action pose is then updated by applying this refinement transformation. This iterative procedure can be executed across multiple stages to enhance accuracy. A visual example of this process can be found

[1] W. Wang and G. Hager are with the Johns Hopkins University, Department of Computer Science, Baltimore, USA. wwang121 @ cs.jhu.edu and hager @ cs.jhu.edu
[2] Y. Lei, S. Jin, L. Zhang are with Baidu Research, Robotics and Autonomous Driving Lab (RAL), Sunnyvale, USA. {yutianhe,shiyujin,liangjunzhang} @ baidu.com
[†] Work done while the author was an intern at Baidu Research, Robotics and Autonomous Driving Lab (RAL), Sunnyvale, USA.

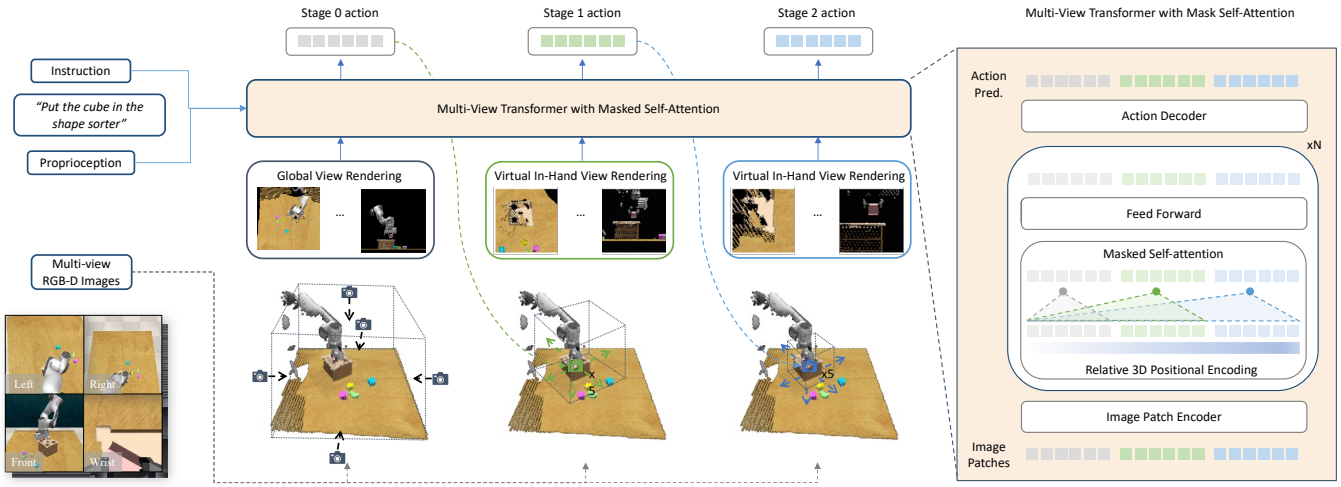

Fig. 3. **VIHE Overview.** Starting with RGB-D images from multi-view cameras, we first construct a point cloud of the scene. Global views are first rendered using fixed cameras positioned around the workspace. From these global views, the network outputs initial action predictions $a^0_{pose}, a^0_{open}, a^0_{col}$. Then at each refinement stage i, we autoregressively generate virtual in-hand views from cameras attached to the previously predicted gripper pose a^{i-1}_{pose}. Based on the rendered views, we then refine the action predictions. The network architecture employs masked self-attention to have tokens from later stages attend to tokens from previous stages. Language instruction tokens are merged into stage 0 image tokens when input into transfer, which is omitted in the figure for conciseness. More information can be found in Sec. III.

in Figure 1. Intuitively, we are asking the model solve a local action refinement problem conditioned on local image context. Such local image context, other than providing higher effective resolution with same image size, is also shown by [10] to be beneficial for generalization as less distracting information is provided to the model. Throughout these stages, we employ masked self-attention mechanisms, allowing tokens from later stages to attend to those from earlier stages and blocking attention in the opposite direction during training. This design effectively integrates information from rendered images across all stages, leading to more accurate final predictions.

Our empirical evaluations affirm the efficacy of incorporating virtual in-hand perspectives into 3D robotic manipulation tasks. We demonstrate that our approach significantly enhances performance across a variety of tasks and settings, thereby providing an effective solution for real-world applications. As shown in Figure 2, our method delivers an 18% improvement in final performance and requires only one-fifth of the training time to achieve performance metrics comparable to existing state-of-the-art methods. In high-precision tasks such as peg insertion, our approach more than triples the success rate when compared to current state-of-the-art methods. Conducted on a Franka robot, we also validate the superiority of our method over baseline approaches through real-world experiments. Lastly, through extensive ablation studies, we validate our design choices, such as rendering from in-hand perspective, zoom-in local image rendering and action prediction by stage-wise refinement, all contribute to final performance improvements.

Our contributions can be summarized below.

1) We introduce a novel representation technique that utilizes virtual in-hand views, and employ a transformer-based network architecture to iteratively refine action predictions based on this representation.
2) We investigate various design choices to efficiently utilize this representation in robotic manipulation tasks.
3) Through empirical evaluation in both simulated and real-world settings, we demonstrate significant improvements in training speed, sample efficiency, and overall performance.

II. Related Work

A. Vision-Based Imitation Learning for Robotic Manipulation

The landscape of imitation learning [11] in robotic manipulation has undergone a significant transformation with the advent of deep learning. Traditional approaches often depended on low-dimensional state observations and engineered features [12], [13], [14]. With the advancement of deep learning [15], [16], studies have explored using convolutional neural networks [17] to have raw 2D RGB images as inputs and predict actions directly [18], [19], [20], [21]. More recently, works such as RT-1 [22], RT-2 [23], GATO [24] and Octo [25] have employed Transformer architectures [26], [27] to predict robot actions directly from high-dimensional multi-modal inputs like images and natural language instructions. However, these methods typically require a large amount of demonstrations for effective learning [22], [23], [24], [25].

The field has also experienced a growing interest in the use of RGB-D images as input, offering richer visual information in the 3D space where manipulation occurs. Methods like CLIPort [28] have shown promise in simple pick-and-place tasks but remain confined to tabletop, top-down scenarios. To fully exploit the 3D information available

in observations, research efforts such as C2F-ARM [29] and PerAct [30] have utilized 3D voxel grids to represent the robot's workspace. While these methods offer spatially precise 3D pose prediction, they come with the drawback of high computational overhead.

Among the most recent advancements [7], [8], [31], [32], [33], the state-of-the-art method Act3D [8] extracts point clouds from RGB-D images and characterizes them using pre-trained CLIP [34] features. Actions are subsequently identified from additionally sampled action proposal points through cross-attention with the previous feature points. Due to the large number of points that need to be sampled for both features and action proposals, Act3D remains computationally demanding, requiring five days of training with only two cross-attention layers. Another recent work, RVT [7] is the most closely related to our own. Like Act3D, RVT also extracts point clouds from RGB-D images but opts to render multi-view orthographic images [35] using these points. The orthographic images provide a useful inductive bias for action location prediction, enabling RVT to achieve performance similar to Act3D but with a significantly reduced training time of just two days. Our method also employs orthographic images as observational input but distinguishes itself by utilizing in-hand views to iteratively refine action pose predictions, setting it apart from RVT and other prior methods.

B. In-Hand View for Robotic Manipulation

Previous research has also highlighted the advantages of an in-hand perspective for robotic manipulation that requires high precision [36], [9], [10], [37], [38], [39]. For instance, [9] demonstrates that an in-hand view reveals more task-relevant details, which is particularly beneficial for high-precision tasks such as peg insertion. Besides precision, [10] also shows that incorporating an in-hand view can mitigate distractions that are irrelevant to gripper actions, thereby improving generalization. Furthermore, [36] builds a data collection system that can transfer human demonstrations to robot policies, leveraging in-hand camera views. However, these studies typically obtain the in-hand perspective through a camera attached to the robot's end-effector, which differs from our approach of rendering in-hand views from point clouds. We further argue that directly using camera-captured in-hand perspectives—when available—would not be as effective in our keypoint-based imitation learning setting, to be elaborated in Sec. III. This is because the next predicted keypoint could be located at a considerable distance from the current hand position (such as in Figure 1), rendering the existing in-hand view less useful.

III. Methods

A. Overview

We investigate the impact of in-hand camera views in the setting of language-conditioned imitation learning with a 3D action and observation space. We assume access to a dataset comprising n demonstration trajectories. Each demonstration consists of a language instruction l, a sequence of m observations $\{{}^0o, \ldots, {}^{m-1}o\}$ and a sequence of m continuous actions $\{{}^0a, \ldots, {}^{m-1}a\}$. In this work, we treat each time step independently, so for simplicity of notation in subsequent developments, we consider a single time step and we use o and a to represent the observation and action at that time. Each observation o includes RGB-D images captured from one or more camera perspectives. An action a consists of a 6-DoF end-effector pose a_{pose} in SE(3), a binary open or closed state a_{open}, and a binary state indicating whether to employ a motion planner to avoid collisions a_{col} while moving to the target pose:

$$a = \{a_{\text{pose}} \in \text{SE}(3), a_{\text{open}} \in \{0, 1\}, a_{\text{col}} \in \{0, 1\}\}. \quad (1)$$

The architecture of the Virtual In-Hand Eye Transformer (VIHE) is depicted in Figure 3. It employs a Transformer-based policy that, at given timestep t, predicts an end-effector pose in SE(3) based on one or more RGB-D images, a language instruction, and proprioceptive state of the robot's current end-effector state. The core concept is to predict and refine the end-effector pose using images rendered according to the predicted pose produced at each stage in an autoregressive process. In line with prior work [7], [8], [30], we identify a set of "keyposes" that capture critical end-effector poses within a demonstration and apply VIHE at only those steps. A pose qualifies as a keypose if: (1) the end-effector changes state (e.g., something is grasped or released), (2) velocities approach zero (commonly observed when entering pre-grasp poses or transitioning to a new phase of a task). The prediction problem is thus reduced to identifying the next action at each keypose based on the current observation. During inference, VIHE iteratively predicts the next best keypose and navigates to it using a motion planner, as in previous works [7], [8], [30].

B. Iterative View Rendering and Action Refinement

In the following, we describe the multi-stage view rendering and action refinement procedure that iteratively predicts and refines action using the multi-view transformer network, F. A visualization of the process can be found in Figure 3.

1) Initial Global Stage: The initial global stage (Stage 0) utilizes a predefined set of fixed virtual camera poses p^0 positioned around a cubic workspace from five directions (top, front, back, left, and right) as in RVT [7]. These poses are accompanied by intrinsic parameters f^0 chosen so that each rendered image that covers the size of the entire workspace. Using a rasterization-based renderer $R(o, p^0, f^0)$, we generate images x^0 from these camera poses and intrinsics. The multi-view transformer network F then predicts the action $a^0_{\text{pose}}, a^0_{\text{open}}, a^0_{\text{col}}$ based on these images x^0 and a given language instruction l. This can be expressed as:

$$x^0 = R(o, p^0, f^0), \quad (2)$$

$$a^0_{\text{pose}}, a^0_{\text{open}}, a^0_{\text{col}} = F(x^0, l). \quad (3)$$

2) Iterative Refinement: In subsequent stages, we generate virtual in-hand view images to provide input for iterative

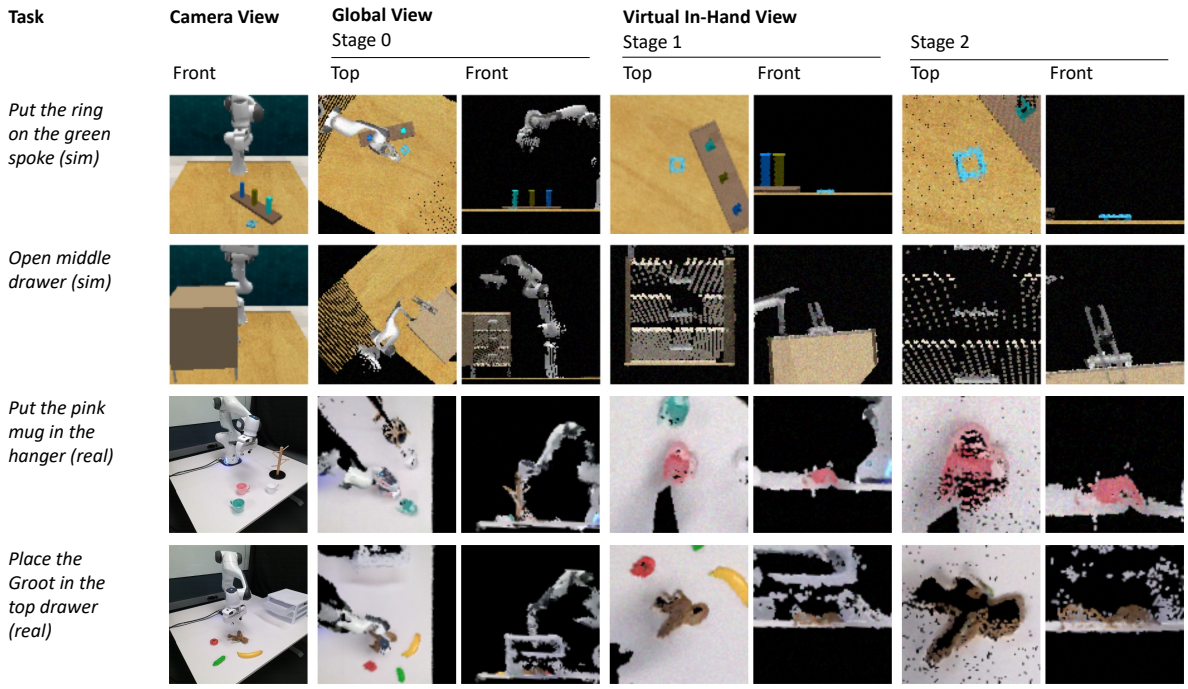

Fig. 4. Visualization of sample images from front camera in RGB view, global view and virtual in-hand view (both rendered as orthographic images). For global orthographic view and virtual in-hand view, two (top and front) out of five views (top, front, back, left and right) are visualized. Virtual in-hand view reveals information that is occluded from global view and in greater details, leading to better manipulation performance.

action refinement. These images are rendered from five cameras with the same relative geometry, but now placed relative to the predicted end-effector pose, effectively serving as "virtual in-hand eyes." To focus on finer details, the intrinsic parameters f^i are adjusted to cover $\frac{1}{2^i}$ of the workspace at each stage i. From $[x^0, \dots, x^i]$ (all rendered images up to stage i) along with language instruction l, we use the network F to predict a transformation refinement h^i_{pose} and updated $a^i_{\text{open}}, a^i_{\text{col}}$. The refinement h^i_{pose} is then applied to the previously predicted action pose a^{i-1}_{pose} to obtain an updated action pose in stage a^i_{pose}. We chose to predict a relative transformation instead of the pose directly to encourage the network to weight information on in-hand views over fixed views in the refinement stages. An ablation study supporting this choice is provided in Sec. IV-A. Let C be a function that maps action poses to camera poses. The iterative refinement can be formalized as:

$$p^i = C(a^{i-1}_{\text{pose}}), \tag{4}$$

$$x^i = R(o, p^i, f^i), \tag{5}$$

$$h^i_{pose}, a^i_{\text{open}}, a^i_{\text{col}} = F([x^0, \dots, x^i], l), \tag{6}$$

$$a^i_{\text{pose}} = h^i_{pose} \circ a^{i-1}_{\text{pose}}. \tag{7}$$

Note that Eq. 3 can be regarded as a special case of Eq. 6 where we refine from base coordinate frame as an intial starting pose.

C. VIHE Architecture

1) Network Architecture: The rendered images $[x^0, \dots, x^2]$, the task language instruction l, and the proprioception data are processed by a transformer network F. In order to ground the instruction in the scene, we use a pre-trained CLIP [40] model to extract a sequence of K_{lang} tokens per instruction. For both fixed and in-hand views, we break each rendered image into K_{img} patches. A one-layer convolutional layer is used to extract in total $5 \times K_{img}$ tokens for each stage that renders 5 views. For the proprioception data, similar to PerAct and RVT, we pass it through an MLP and concatenate it to the image tokens. We repeat refinement two times - we have 1 initial global stage and 2 refinement stages.

To efficiently train action prediction for all stages, VIHE has L masked self-attention layers [26] following the patch encoding layer. Given the set of all $3 \times 5 \times K_{img} + K_{lang}$ tokens across all 3 stages, we construct a mask M so that each token can attend to other tokens within the same or earlier stages. Language tokens are treated as stage 0 tokens to allow all stage tokens to access instruction information. The masked self-attention is then formulated as:

$$F_{\text{attention}} = \text{Softmax}\left(\frac{QK^T + M}{\sqrt{d_k}}\right)V \tag{8}$$

where Q, K, V denotes query, key and value vectors in dimension d_k in each attention layer.

To efficiently fuse information from multi-views captured from different camera poses in 3D, we apply the Rotary Position Embedding (RoPE) [41] to the query and key when calculating the attention scores. For each image patch token, we compute its position projected to the camera image plane and use that 3D location to compute RoPE embedding.

This is similar to Act3D [8] where they apply RoPE to cross-attention over points. For all language tokens, we directly learn a RoPE embedding function since no location is relevant to the language. We additionally add learned positional embedding to language tokens and learned view-specific embedding to image tokens.

2) Action Prediction: In initial stage 0, the transformer network F outputs 8-dimensional action composing of a_{pose}, a_{open} and a_{col}. In action prediction, we predict the 6-DoF a_{pose} in SE(3) as a 3-DoF translation component a_{trans} and a 3-DoF rotation component a_{rot}. As in PerAct [30], a_{rot} is represented as Euler angles discretized into 5 degree bins. Following RVT [7], We first predict a heatmap for each view. Heatmaps from all views are then projected to predicted score for a set of 3D points to select a_{trans} with the best score. Using the same set of heatmaps, we predict a_{rot}, a_{open} and a_{col} from image features weighted by the predicted translation heatmap.

In subsequent refinement stages, the network outputs refinement transformation h_{pose} in place of a_{pose}. We note that, because the initial pose is from the base coordinate system, it is in fact a refinement from the world (identity) pose. We use the final stage outputs as the prediction of our model.

3) Training: Our model predicts later stage actions conditioned on the prediction from earlier stages. This makes our model inference autoregressive in nature. In autoregressive models, exposure bias [42] is known to deteriorate the performance due to the error accumulation from the discrepancy of predictions in training and inference. To mitigate this issue, we use stochastic sampling which adds a random perturbation when sampling camera poses for the refinement stages during training. Specifically, we add $\epsilon_i \in \mathrm{SE}(3)$ to groundtruth a_{pose} when rendering in-hand view images x_i for stage i. This is different from testing where we use the predicted a_{pose}^{i-1} to render in-hand view images as described in the previous subsection.

Our final action prediction loss function is a summation of individual loss functions across all stages. Similar to PerAct [30] and RVT [7], we choose to use cross-entropy loss for each component. For the continuous component a_{trans}, a target heatmap is obtained by applying a truncated Gaussian kernel to the 2D projection location of groundtruth 3D action as in RVT. The cross-entropy loss is then computed between the target heatmap and the predicted heatmap for each view. The final training loss $\mathcal{L}_{action}$ is then the summation of individual losses across all three stages as below.

$$\mathcal{L}_{action} = \sum_{i=0}^{2} [\mathcal{L}_{pose}^{i} + \mathcal{L}_{open}^{i} + \mathcal{L}_{col}^{i}]. \tag{9}$$

where $\mathcal{L}_{pose}^{i} = \mathcal{L}_{trans}^{i} + \mathcal{L}_{rot}^{i}$. L_{trans}^{i}, L_{rot}^{i}, L_{open}^{i} and L_{col}^{i} are losses for each component respectively.

IV. Experiments

We evaluate VIHE in a multi-task, language-conditioned, vision-based imitation learning setting in both simulation and the real world. We conduct our simulated experiments in RLBench [43], an established simulation benchmark for learning manipulation policies. To validate its real-world applicability, we further test it on a physical Franka Panda arm [44].

A. Simulation Experiments

Simulation Setup. Our experimental framework closely aligns with the one established in PerAct [30]. We utilize CoppeliaSim [45] to simulate a range of tasks from RLBench [43], employing a Franka Panda robot equipped with a parallel gripper. We assess performance across the same 18 tasks introduced by PerAct, which encompass a diverse set of activities such as pick-and-place, tool manipulation, and high-precision peg insertions. Each task is further diversified through variations guided by associated language descriptions. Visual observations are acquired from four 128×128 resolution RGB-D cameras. To compute the target gripper pose, we employ a sampling-based motion planner [46], consistent with prior work [7], [8], [30].

Baselines. We benchmark our approach against five established baselines: (1) Image-BC [47], which is an image-to-action behavior cloning agent with CNN and ViT vision encoder variants; (2) C2F-ARM-BC [29], which transforms RGB-D images into multi-resolution voxels and predicts key-frame actions using a coarse-to-fine approach; (3) PerAct [30], which encodes RGB-D images into voxels and uses a Perceiver [48] transformer for action prediction; (4) RVT [7], which renders five global orthographic images from input RGB-D images and uses a multi-view transformer for action prediction; and (5) Act3D [8], which uses pre-trained image features and relative cross-attention on point clouds to detect actions.

Implementation Details. We use the pre-trained ResNet-50[49] variant of the CLIP model to acquire a sequence of $K_{\text{lang}} = 77$ tokens per instruction. We utilize the same image rendering pipeline introduced by RVT [7]. RGB-D camera data is first projected into a point cloud, which is then used to render orthographic images from given viewpoints. Each rendered view consists of three image maps with a total of seven channels: 3 channel RGB, 1 channel depth, and 3 channels world-frame coordinates. All virtual images are rendered with a resolution of 110×110. Each image is divided into $K_{\text{img}} = 100$ patches of size 11×11. The model uses $L = 8$ layers of masked self-attention, with an input of $K_{\text{lang}} + 3 \times 5 \times K_{\text{img}} = 1577$ tokens. Notably, we render virtual views at half the resolution compared to RVT, yet achieving superior performance. This enhancement is attributable to the coarse-fine refinement strategy that capitalizes on the inherent multi-scale information in the images, facilitating more nuanced adjustments and optimization in the refinement stages.

Training and Evaluation. We follow the training protocol that aligns with previous works [30], [7] for fair benchmarking. We utilize the same RLBench dataset introduced by PerAct [30], along with the same data augmentation techniques. The model is trained for 150,000 steps using the LAMB optimizer [50], with a batch size of 24 and an

TABLE I

MULTI-TASK PERFORMANCE ON SIMULATED BENCHMARK.

Models	Avg. Success		Training Time (in Days)	Close Jar		Drag Stick		Insert Peg		Meat off Grill		Open Drawer		Place Cups		Place Wine		Push Buttons	
	10	100		10	100	10	100	10	100	10	100	10	100	10	100	10	100	10	100
Image-BC (CNN)	2	1	-	0	0	0	0	0	0	0	0	4	4	0	0	0	0	4	0
Image-BC (ViT)	4	1	-	0	0	0	0	0	0	0	0	16	0	0	0	4	0	16	0
C2F-ARM-BC	23	26	-	28	24	72	24	0	4	40	20	28	20	0	0	36	8	88	72
PerAct	30	43	16	32	60	36	68	4	0	68	84	68	80	0	0	20	12	56	48
RVT	45	60	1.7	52	44	**100**	**100**	8	12	**72**	92	84	76	0	4	60	56	72	**100**
Act3D	48	65	5	**90**	**92**	52	92	7	27	58	94	**92**	**93**	1	3	32	80	**98**	99
VIHE (ours)	**57**	**77**	**1.6**	36	48	**100**	**100**	**24**	**84**	**72**	**100**	72	76	**4**	**12**	**60**	**88**	88	**100**

Models	Put in Cupboard		Put in Drawer		Put in Safe		Screw Bulb		Slide Block		Sort Shape		Stack Blocks		Stack Cups		Sweep to Dustpan		Turn Tap	
	10	100	10	100	10	100	10	100	10	100	10	100	10	100	10	100	10	100	10	100
Image-BC (CNN)	0	0	0	8	0	4	0	0	4	0	0	0	0	0	0	0	0	0	20	8
Image-BC (ViT)	4	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	8	0	24	16
C2F-ARM-BC	4	0	12	4	0	12	12	8	12	16	8	8	4	0	0	0	4	0	60	68
PerAct	0	16	16	68	16	44	28	24	32	72	16	20	12	36	0	0	72	56	72	80
RVT	16	52	60	92	44	56	16	52	48	**100**	8	40	0	36	4	20	76	72	**96**	76
Act3D	**27**	51	82	90	**75**	**95**	26	47	**66**	93	7	8	6	12	8	9	82	**92**	64	**94**
VIHE (ours)	24	**60**	**96**	**96**	52	92	**60**	**92**	60	96	**32**	**52**	**32**	**68**	**24**	**68**	**100**	64	92	92

TABLE II

ABLATION STUDY ON SIMULATED BENCHMARK.

Variants	Avg. Succ.	Perf. Change
Full VIHE model	77	-
Network architectures		
No cross-stage attention	69	-8
Direct predict a_{pose} in refinement	71	-6
No rotary positional encoding	74	-3
Virtual in-hand view rendering		
No zoom-in for virtual in-hand views	63	-14
Fixed rotation in virtual in-hand views	69	-8
Position camera to look outside-in	75	-2
Inference		
Use stage 0 action predictions for inference	50	-27
Use stage 1 action predictions for inference	71	-6

initial learning rate of 2.4×10^{-3}. We train our method using the same hardware of 8 NVIDIA Tesla V100 GPUs as in previous works [30], [7], [8]. The model is tested on the same 25 variations for each task as in PerAct [30]. Benchmark evaluation in Table I leverages results from respective sources for Image-BC [47], C2F-ARM-BC [29], PerAct [30], and Act3D [8]. We retrain RVT [7] using author released code for fair comparison.

Results. As presented in Table I, VIHE surpasses all baselines in success rate when averaged across all tasks. With 100 demonstrations, it outperforms the existing SOTA method RVT by 17 percentage points (a 28% relative improvement) and Act3D by 12 percentage points (an 18% relative improvement). Our method's advantage sustains when only 10 demonstrations are provided, outperforming Act3D by 9 percentage points (a 19% relative improvement). Remarkably, our performance using 10 demonstrations per task is close to the performance using 100 demonstrations per task for RVT. These results demonstrate that VIHE is both more accurate and more sample-efficient compared to existing state-of-the-art methods. The improvements mainly come from large gains in challenging high-precision tasks. For instance, VIHE greatly improves the success rate on the "Insert Peg" task from 27 to 84, more than tripling its performance and highlighting our method's capability in performing challenging high-precision tasks.

Ablation Study. As shown in Table II, we conduct extensive ablation experiments on the same multi-task environments under 100 demonstrations. We analyze different design choices of VIHE , grouped into three categories: architecture, in-hand view rendering configurations, and model inference.

Architecture: We investigate the effect of different network components. Without cross-stage attention, the performance drops by 8%. Directly predicting a_{pose} instead of predicting the relative transformation h_{pose} results in a performance decrease of 6%. Rotary positional encoding also helps to efficiently relate information from patches in different stages, resulting in a 3% difference.

Virtual In-Hand View Rendering: We vary camera settings when performing virtual in-hand view rendering. We found that iteratively decreasing the field of view by half in three stages is important, without which the performance drops to 63%. Meanwhile, we also want to highlight that even without such iterative zoom-in, our model still largely outperforms single stage prediction, which achieves a 50% success rate using images of our 100 resolution and 60% using larger 220 resolution from RVT. This indicates that the in-hand pose itself, independent from iteratively zoom-in, offers strong inductive-bias for model to correlate image and action refinement. Additionally, we also test rendering with fixed rotation and outside-in camera positions and find that they lead to performance drops of 8% and 2%, respectively.

Inference: A model with only global observations at stage 0 suffers a 27% performance drop, and a model with one additional refinement at stage 1 suffers a 6% performance drop. This demonstrates that all three stages are important for achieving high performance.

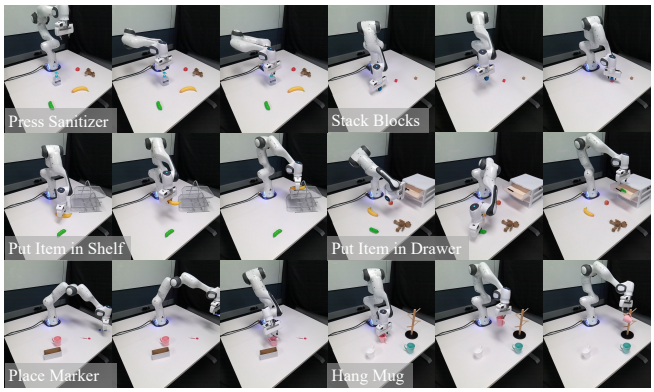

Fig. 5. **Real-world object manipulation tasks.** A single VIHE model can perform multiple tasks (6 tasks, 18 variations) in the real world with just 72 demonstrations in total.

B. Real-World Experiments

Real-World Setup. We experiment on a table-top setup using a Franka Panda arm [44]. The scene is perceived via an Azure Kinect (RGB-D) camera [51] statically mounted in a third-person view. We design a total of 6 language-conditioned tasks with 18 variations, including different items, colors, and targets. A total of 72 demonstration trajectories are collected by executing human-specified waypoints with random locations of objects. Visual examples of our tasks can be found in Figure IV-A. More videos can be viewed on our project website.

TABLE III
MULTI-TASK CONFIGURATION AND PERFORMANCE ON REAL ROBOT.

Task	Variation	# Train	RVT Succ.	VIHE Succ.
Press Sanitizer	N/A (1)	5	9/10	**10/10**
Stack Blocks	color (6)	12	4/10	**7/10**
Put in Shelf	item (2)	10	6/10	**9/10**
Put in Drawer	item (2)	10	3/10	**5/10**
Place Marker	color and target (4)	20	2/10	**6/10**
Hang Mug	color (3)	15	1/10	**7/10**
All tasks	18	72	25/60	**43/60**

Results. We train our VIHE and the baseline method RVT on real-world data for 50,000 steps, using the same hyper-parameters as in the simulated environment. Success rates across various tasks are provided in Table III. Overall, VIHE achieves a significant success rate of 43/60, outperforming RVT by a large margin. The results demonstrate our method's effectiveness in real-world 3D object manipulation tasks. Especially, the performance of VIHE stands out in tasks demanding high precision, such as "Place Marker" and "Hang Mug".

V. CONCLUSION

In this work, we introduced a novel approach for robotic manipulation that leverages virtual in-hand views to iteratively refine action predictions. Our method offers an observation space that significantly enhances performance in 3D object manipulation. Through empirical evaluations, we showed that our approach substantially outperforms existing SOTA methods in both simulated environments and physical robots. In extensive ablation studies, we highlighted the importance of various design choices, including in-hand view rendering, relative transformation prediction, cross-stage attention, and zoom-in camera views.

We also identify some limitations that present avenues for future research. Similar to prior methods, VIHE requires calibrated RGB-D cameras to obtain point clouds for image rendering. Integrating our approach with NeRF-like implicit view rendering techniques [32], [52] may eliminate this constraint and broaden its applicability. Additionally, our current model does not utilize pre-trained image features, the inclusion of which may further improve performance.

REFERENCES

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.
[2] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of machine learning research*, vol. 22, no. 30, pp. 1–82, 2021.
[3] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.
[4] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International Conference on Machine Learning*. PMLR, 2022, pp. 9118–9147.
[5] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.
[6] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 785–799.
[7] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox, "Rvt: Robotic view transformer for 3d object manipulation," *arXiv preprint arXiv:2306.14896*, 2023.
[8] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: Infinite resolution action detection transformer for robotic manipulation," *arXiv preprint arXiv:2306.17817*, 2023.
[9] R. Jangir, N. Hansen, S. Ghosal, M. Jain, and X. Wang, "Look closer: Bridging egocentric and third-person views with transformers for robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3046–3053, 2022.
[10] K. Hsu, M. J. Kim, R. Rafailov, J. Wu, and C. Finn, "Vision-based manipulators need to also see from their hands," *arXiv preprint arXiv:2203.12677*, 2022.
[11] B. Zheng, S. Verma, J. Zhou, I. W. Tsang, and F. Chen, "Imitation learning: Progress, taxonomies and challenges," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
[12] C. C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments [grand challenges of robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 20–29, 2007.
[13] J. Bandera, J. Rodriguez, L. Molina-Tanco, and A. Bandera, "A survey of vision-based architectures for robot learning by imitation," *International Journal of Humanoid Robotics*, vol. 9, no. 01, p. 1250006, 2012.
[14] R. M. Murray, Z. Li, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 2017.
[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[17] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[18] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4693–4700.

[19] A. Wang, T. Kurutach, K. Liu, P. Abbeel, and A. Tamar, "Learning robotic manipulation through visual planning and acting," *arXiv preprint arXiv:1905.04411*, 2019.

[20] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto, "The surprising effectiveness of representation learning for visual imitation," *arXiv preprint arXiv:2112.01511*, 2021.

[21] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," in *Conference on Robot Learning*. PMLR, 2021, pp. 1992–2005.

[22] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[23] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[24] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, *et al.*, "A generalist agent," *arXiv preprint arXiv:2205.06175*, 2022.

[25] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, D. Sadigh, C. Finn, and S. Levine, "Octo: An open-source generalist robot policy," https://octo-models.github.io, 2023.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[27] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI open*, vol. 3, pp. 111–132, 2022.

[28] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 894–906.

[29] S. James, K. Wada, T. Laidlow, and A. J. Davison, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 739–13 748.

[30] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 785–799.

[31] P. Parashar, J. Vakil, S. Powers, and C. Paxton, "Spatial-language attention policies for efficient robot learning," *arXiv preprint arXiv:2304.11235*, 2023.

[32] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang, "Gnfactor: Multi-task real robot learning with generalizable neural feature fields," *arXiv preprint arXiv:2308.16891*, 2023.

[33] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3d diffuser actor: Policy diffusion with 3d scene representations," *arXiv preprint arXiv:2402.10885*, 2024.

[34] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[35] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.

[36] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *arXiv*, 2024.

[37] E. Y. Puang, K. P. Tee, and W. Jing, "Kovis: Keypoint-based visual servoing with zero-shot sim-to-real transfer for robotics manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7527–7533.

[38] S. Song, A. Zeng, J. Lee, and T. Funkhouser, "Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4978–4985, 2020.

[39] E. Valassakis, N. Di Palo, and E. Johns, "Coarse-to-fine for sim-to-real: Sub-millimetre precision across wide task spaces," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5989–5996.

[40] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[41] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *arXiv preprint arXiv:2104.09864*, 2021.

[42] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," *arXiv preprint arXiv:1511.06732*, 2015.

[43] S. James, A. J. Davison, and E. Johns, "Rlbench: The robot learning benchmark & learning environment," in *Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9616–9622.

[44] K. Zhang, M. Sharma, J. Liang, and O. Kroemer, "A modular robotic arm control stack for research: Franka-interface and frankapy," *arXiv preprint arXiv:2011.02398*, 2020.

[45] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1321–1326.

[46] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[47] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 991–1002.

[48] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, *et al.*, "Perceiver io: A general architecture for structured inputs & outputs," *arXiv preprint arXiv:2107.14795*, 2021.

[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[50] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, "Large batch optimization for deep learning: Training bert in 76 minutes," *arXiv preprint arXiv:1904.00962*, 2019.

[51] M. Tölgyessy, M. Dekan, L. Chovanec, and P. Hubinskỳ, "Evaluation of the azure kinect and its comparison to kinect v1 and kinect v2," *Sensors*, vol. 21, no. 2, p. 413, 2021.

[52] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.