2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Evaluating Robustness of Visual Representations for Object Assembly Task Requiring Spatio-Geometrical Reasoning

Chahyon Ku[1], Carl Winge[1], Ryan Diaz[1], Wentao Yuan[2] and Karthik Desingh[1]

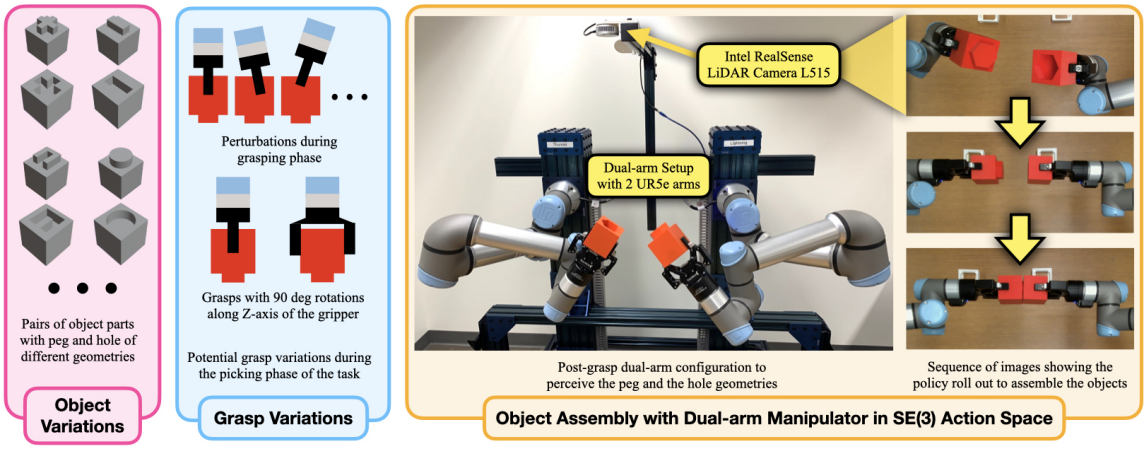

Fig. 1: An overview of our benchmarking setup. Benchmarking robustness under object variations (left) and grasp variations (center) of visual policy learning methods on object assembly task with a dual-arm manipulator in SE(3) action space (right)

***Abstract*— This paper primarily focuses on evaluating and benchmarking the robustness of visual representations in the context of object assembly tasks. Specifically, it investigates the alignment and insertion of objects with geometrical extrusions, commonly referred to as a peg-in-hole task. The accuracy required to detect and orient the peg and the hole geometry in SE(3) space for successful assembly poses significant challenges. Addressing this, we employ a general framework in visuomotor policy learning that utilizes visual pretraining models as vision encoders. Our study investigates the robustness of this framework when applied to a dual-arm manipulation setup, specifically to the grasp variations. Our quantitative analysis shows that existing pretrained models fail to capture the essential visual features necessary for this task: a visual encoder trained from scratch consistently outperforms the frozen pretrained models. Moreover, we discuss rotation representations and associated loss functions that substantially improve policy learning. We present a novel task scenario designed to evaluate the progress in visuomotor policy learning, with a specific focus on improving the robustness of intricate assembly tasks that require both geometrical and spatial reasoning. Videos, additional experiments, dataset, and code are available at https://sites.google.com/view/geometric-peg-in-hole.**

I. INTRODUCTION

Peg-in-hole assembly has been a longstanding and actively researched problem within the field of robotics as an essential component of industrial robotics that require reasoning over geometries, affordances, and contact models of objects. Prior works have predominantly focused on single-arm top-down peg-in-hole tasks, where the hole is attached to a fixture, while a robot arm picks and inserts the peg in top-down action space [1]–[8]. While these approaches effectively reduce the complexity of the task to focus on contact modelling and sub-millimeter-precision for industrial use cases, they make assumptions unfit for robot manipulation in more unstructured environments. For robots to adequately manipulate household items with parts that fit together (e.g. containers and lids), we consider the following requirements to be essential. To manipulate a diverse set of containers and lids the robot will encounter, the robot should be able to look at and reason about the intra-category variations of geometry (food containers vs. water bottles) as well as the objects' spatial (what is the relative transformation between parts?) and geometric (what parts fit together in what way?) relationship. To operate in unstructured indoor environments, the robot may not assume objects are fixed to a flat surface but rather pick up both relevant parts and assemble them using two arms in full SE(3) action space. The robot also may not have multiple external cameras giving a near-complete third-person-views of the task, the robot, and the objects.

Based on these requirements, we propose a novel geometric-peg-in-hole task, inspired by traditional peg-in-hole but adapted for object variations and unstructured environments, that evaluates the spatio-geometric reasoning capabilities of visuo-motor policies. Our pegs and holes have geometric variations in the extrusions such as plus-sign, minus-sign, or pentagon, which require the robot to

This project is supported by the Minnesota Robotics Institute Seed Grant, Undergraduate Research Scholarships, and Undergraduate Research Opportunities Program at the University of Minnesota.

[1] C. Ku, C. Winge, R. Diaz, and K. Desingh are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 US. {ku000045, winge134, diaz0329, kdesingh}@umn.edu

[2] W. Yuan is with the Paul. D Allen School of Computer Science and Engineering, University of Washington, Seattle, WA 98195 US. wentaoy@cs.washington.edu

generalize over these intra-category variations without prior knowledge, as well as the different amounts of rotational variation required to put them together (see Fig. 3 with rotational symmetries). The task starts out with the peg and hole in each gripper, but with randomized grasp noise to test the policy's ability to generalize over uncertainties during grasp estimation and execution (see Fig. 1 center). As both objects are held and controlled by the arms, the robot should not only reason about the current and goal SE(3) transformations between the objects, but also make use of all axes of SE(3) action space to insert the objects. On top of the simulation environment created using PyBullet [9], we also design a repeatable task setup in the real world, where the objects are always picked up and put down in the same position and orientation on a fixture, such that data collection and policy evaluation can be done without human intervention (see supplementary video and Fig. 1 right).

Then, how can we train a robot to complete such a task? End-to-end learning from demonstrations offers a flexible framework for training robust robots without the need to explicitly model object and task representations. Taking wisdom from the recent success of pre-trained representations in natural language processing and computer vision, recent works such as R3M [10] and MVP [11] propose robotics-specific visual representations trained from large-scale datasets of ego-centric videos. To assess the geometric reasoning capability of visual representations, we conduct a performance comparison. This comparison involves evaluating the performance of from-scratch trained baselines against various pre-trained representations in a image-to-action imitation learning setup.

In summary, our main contributions are:

- A novel dual-arm geometric-peg-in-hole task designed to evaluate spatio-geometric reasoning capabilities of visual representations on 9 pairs of peg and hole objects with randomized grasp in rotation and translation of all axes.
- A comprehensive analysis of eight visual representations, including two from-scratch encoders and six pre-trained models, is conducted to train policies within an imitation learning framework. This assessment quantitatively measures their success in handling increasingly challenging task (grasp) variations with two different imitation learning methods.
- An evaluation of control representations (absolute vs. delta) and rotation representations (quaternion vs. ortho6d) in the context of our task learning.
- A simulation environment, dataset, trained models, and evaluation code for others to benchmark and extend upon this task.

II. RELATED WORKS

A. Imitation Learning for Assembly Tasks

End-to-end learning has demonstrated that robots can learn to control their actuators directly from raw sensor observations [12]–[15], avoiding the need for a brittle modular pipeline with explicit state estimation. Traditionally, imitation learning works such as Form2Fit [16] and Transporter [17] formulated assembly tasks as a supervised classification task of predicting SE(2) pick-and-place locations to train high-performing and data-efficient imitation learning policies. These methods require a tabletop setup with a clear camera-view of the objects, stable insertion targets, and 2 dimensional action spaces. More recently, Robomimic [18] proposed an imitation learning benchmark with SE(3) action space, and trained imitation learning policies for assembly-like tasks such as Tool Hang (insert hook onto base frame, hang tool on hook) and Square (place square nut on a rod). While these tasks featured SE(3) action space and rotation of grasped objects, they focused on a single set of objects without geometric variation and on single-arm top-down insertion. In this work, we propose a dual-arm SE(3) insertion task that evaluates the robot's ability to generalize over object variations and reason about SE(3) transformations between the parts.

B. Pretraining for Robotics

To avoid retraining the neural network for every task, more recent works have leveraged state-of-the-art visual pretrained models such as ResNet [19] pretrained on ImageNet [20], CLIP [21], MAE [22], and R3M [10] to obtain visual features to train a policy network. CLIP [21] achieves state-of-the-art results in tasks like ImageNet classification by performing contrastive pretraining on a large-scale non-public image-caption dataset. R3M [10] utilizes time contrastive loss and video-language alignment to pretrain a ResNet [19] on egocentric videos sourced from the Ego4D dataset [23]. MAE [22] achieves state-of-the-art transfer results in image classification by pretraining visual transformers [24] on the ImageNet data. The pretraining process involves using masked input reconstruction, also known as masked autoencoding, as the supervision signal. MVP [11] utilizes masked autoencoding from MAE to pretrain vision transformers [24] on egocentric videos sourced from a combination of datasets, such as Epic Kitchens [23]. In this work, we compare Non-pretrained ResNet-18 and ResNet-50 models with the above mentioned pretrained models except for MVP (does not provide ViT/B or ResNet-50 variants others commonly have).

III. IMITATION LEARNING FRAMEWORK

The goal of imitation learning is to train a policy $\pi : \mathcal{O} \rightarrow \mathcal{A}$ that maps all observations to an action that will progress the robot towards executing the task. Our dataset $\mathcal{D} = \{(O_i, a_i)\}_{i=1,...N}$ consists of N observation-action pairs. Each observation $O_i = ((I_v)_{v \in V}, s)$ is a tuple of RGB images I_v from view $v \in V$ and the current state of the robot $s \in \mathcal{S}$. Each action $a_i \in \mathcal{A}$ is the action the expert performs during demonstration.

Our architecture shown in Fig. 2 is inspired by imitation learning evaluation frameworks from Robomimic [18], R3M [10], and MVP [11]. In our implementation, the policy π_θ consists of three parts: the image preprocessor f, the image encoder g_ϕ, and the policy head h_ψ. The image preprocessor $f : \mathcal{I}^{640\times480} \rightarrow \mathcal{I}^{224\times224}$ crops and resizes the original RGB image to a consistant size for fair comparison

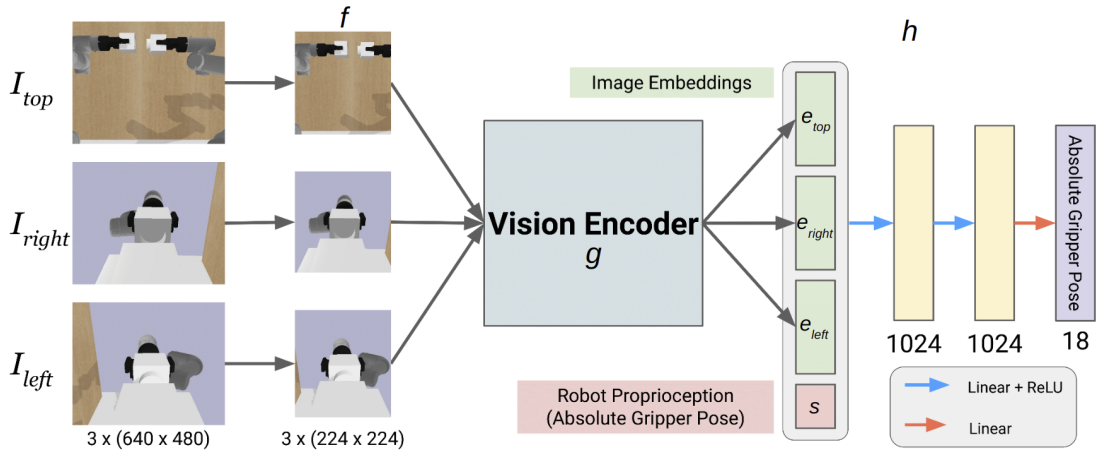

Fig. 2: Network architecture used in the imitation learning pipeline. The network receives 3 images as input, produces image embeddings for each image using a vision encoder of choice (pretrained or non-pretrained), concatenates them all with robot proprioception, and outputs the next action through an MLP head.

of all vision encoder models (ViT-B/16 requires this size). The image encoder $g_\phi : \mathcal{I}^{224\times224} \rightarrow \mathbb{R}^D$ is a neural network that deterministically maps an RGB image I_v to a D-dimensional image embedding e_v. The policy head $h_\psi : \mathbb{R}^D \times ... \times \mathbb{R}^D \times \mathcal{S} \rightarrow \mathcal{A}$ is a multi-layer perceptron on top of concatenated image embeddings $(e_v)_{v\in V}$ and robot state s which produces the final output action a. In summary, output of the policy $\pi_\theta(O) = h_\psi(g_\phi(f(I_{v_1})), ..., g_\phi(f(I_{v_{|V|}})), s) = \hat{a}$ is the predicted action. We train either the parameters ψ (frozen g) or both ϕ and ψ (unfrozen g) by back-propagating the mean squared error loss $\mathcal{L} = MSE(a, \hat{a})$.

We choose $\mathcal{S} \subseteq SE(3)\times SE(3)$ and $\mathcal{A} \subseteq SE(3)\times SE(3)$ to be absolute gripper poses of both arms. For each arm, there are 3 values representing xyz position and 6 values representing the first two columns of the rotation matrix [25], so we represent both as a 18 dimensional vector. The reason for this choice is explained in Sec. V.

IV. TASK SETUP

A. Objects

We generate 3D models of nine peg and hole object pairs using Blender [26] (Fig. 3). To evaluate understanding of geometry rather than precision, we design all pairs to assemble with 1cm tolerance, which allows for a rotation of maximum 5°. Each object pair has one object (the "peg") with an extrusion of a specific geometrical shape, with the other object (the "hole") having an equivalently-shaped hole. The objects are categorized based on their rotational symmetries (i.e. of the orders 1, 2, and 4). When rotating the object by $\{0°, 90°, 180°, 270°\}$, order 1 objects have 4 unique orientations (see Fig. 3, top row), order 2 objects have 2 unique orientations, and order 4 objects have 1 unique orientation. The objects used in the real-world environment are 3D printed directly from the generated 3D models.

B. Simulation Setup

The PyBullet [9] simulation environment (see Fig. 4) is used for data collection and imitation learning evaluation.

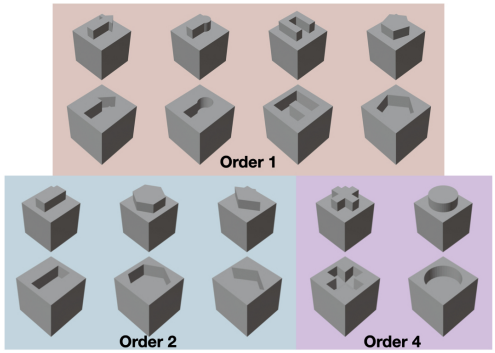

Fig. 3: 9 peg and hole pairs with various shapes, such as *arrow*, *key*, *U*, *pentagon*, *minus-sign*, *hexagon*, *diamond*, *plus-sign*, and *circle*, grouped into categories with rotational symmetries of orders 1, 2, and 4.

It consists of the calibrated clone of our real world setup including two 6-DoF robot arms (Universal Robots UR5e), with two parallel jaw grippers (2F-85 Robotiq grippers) and one top-down view RGB camera (Intel RealSense LiDAR Camera L515 in the real-world). Additionally, two wrist RGB cameras are added in the simulation setup to experiment if multiple views help with the policy learning.

Task Initialization: In the beginning of the episode, peg and hole are randomly held in the gripper based on the specific task variation. As shown in Fig. 5, the in-hand pose of the cap and bottle are randomized from the base pose in 4 ways: XT is the noise in the X-axis Translation of the gripper, uniformly sampled from [-0.01m, 0.01m]. ZT is the noise in the Z-axis Translation of the gripper, uniformly sampled from [-0.01m, 0.01m]. YR is the noise in the Y-axis Rotation of the gripper, uniformly sampled from [-11.25°, 11.25°]. ZR is the random Z-axis Rotation of the gripper, uniformly sampled from {0°, 90°, 180°, 270°}. Note that ZR is not noise but rather a discrete set of rotations that requires the model to conduct geometric reasoning, whereas XT, ZT, and YR are noisy perturbations.

Scripted Expert Demonstrations: The task starts in the "show" state (see Fig. 4): a fixed joint configuration that

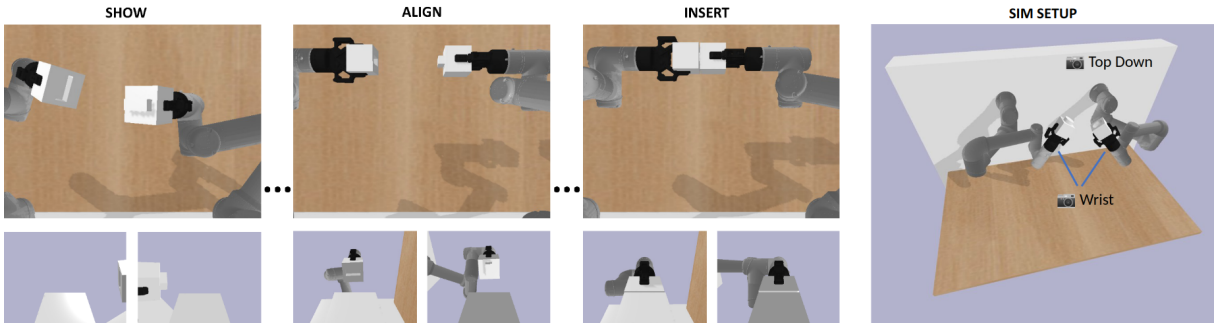

Fig. 4: Task execution sequence in the simulation setup with top-view and left and right wrist-views.

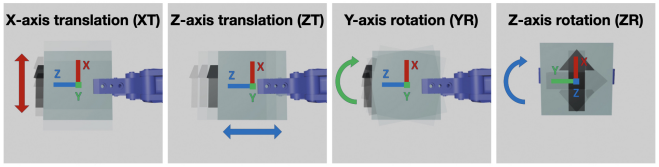

Fig. 5: Possible variations in grasping the objects.

points both grippers towards the camera. With the objects in hand, both arms simultaneously move towards the "align" pose (see Fig. 4), where the objects are perfectly aligned with each other. Afterwards, the arms make a simple linear cartesian move towards the "insert" pose (see Fig. 4), where the objects are put together.

Demonstration Statistics: Roughly 44% of the demonstrations have order 1 objects, 33% have order 2 objects, and 22% have order 4 objects due to the distribution of possible shapes in each category (see Fig. 3). Each demonstration contains a minimum of 10 frames and a maximum of 40 frames. The task is considered successful if the relative transform of the peg and the hole are apart within a margin of 1cm in translation and 5° in orientation from the expected ground-truth relative transform of the peg and the hole. While this is a relaxation of the precision required for industrial peg-in-hole tasks, it still poses significant challenges to vision-based systems, as SOTA algorithms such as GDR-Net [27] are evaluated on 2cm accuracy for tabletop object pose estimation.

C. Real World Setup

The real environment (as seen in Fig. 1 right) is identical to the simulation environment with a few additions to allow for automatic data collection and evaluation. Since objects cannot simply be spawned into the real-life grippers like in simulation, we create a picking phase before the episode and a placing phase after the episode. After the blocks are picked, the robot moves to a predefined joint state to show the blocks to the camera. From there, the robot runs either the scripted demonstration or rollout of the trained policy as in the simulation. After the execution, the system always places the peg and hole in a fixed pose such that the extrusion and intrusion are face-down from the top-view inside 2 3D-printed fixtures to ensure precise placement. Thus, the system can always assume that the peg and hole are picked up from the same pose and can vary the gripper-object transformation as needed for each task variation. For more details on the real world setup, refer to the supplementary video.

V. EXPERIMENTS

A. Experimental Setup

Task Variations: We compare performance on the four grasp variations (XT, ZT, YR, ZR) shown in Fig. 5 and their combinations XTZR, ZTZR, YRZR, and XZTYZR (denoting all variations). The ZR variation requires the model to perform geometric-spatial reasoning to succeed, whereas the other variations require the model to be robust to perturbations in the grasping. Hence, complex variation compositions involving ZR requires the model to do geometric-spatial reasoning to perform the assembly task successfully.

Model Variations: We compare the performance of 2 randomly initialized image encoders and 6 pretrained frozen image encoders (3 ResNet [19] models and 3 vision transformer [24] models). Non-pretrained ResNet-18 and ResNet-50 are randomly initialized and jointly trained with the policy head. ImageNet ResNet-50, R3M ResNet-50, and CLIP ResNet-50 are frozen ResNet-50s initialized with ImageNet [20], R3M [10], and CLIP [21] weights. ImageNet ViT-B/16, CLIP ViT-B/16, and MAE ViT-B/16 are frozen ViT-B/16s initialized with ImageNet, CLIP, and MAE [22] weights.

We use ResNet-50 and ViT-B/16 variants of the pretrained models, because they are the only versions available for all above-mentioned pretraining schemes and have been used for imitation learning in previous works [10], [11].

Object Set Variations: As shown in Fig. 3, we have objects with rotational symmetries of Order-1, Order-2, and Order-4. Order-all denotes the union of Order-1, Order-2, and Order-4, which is used for training all models.

B. Simulation Results

In this section, we will compare the performance of models trained and tested with simulation. Unless otherwise indicated, we use Non-pretrained ResNet-18. **Note:** All models in the simulation experiments are trained with datasets containing all 9 objects from Order-1, Order-2, and Order-4 to test the models' capability to generalize over geometric variations.

Comparison of Visual Representations: All visual representation, pretrained or non-pretrained, are mostly successful

in task variants with just grasp noise perturbations, with success rates of 0.900 or above. However, we observe that the non-pretrained models perform best on ZR that requires geometric-spatial reasoning, with average 0.800 success rates, compared to the best performing pretrained models (CLIP) with average 0.600 success rate (TABLE I).

	XT	ZT	YR	ZR
Non-pretrained ResNet-18	**1.000**	**1.000**	**1.000**	**0.775**
Non-pretrained ResNet-50	**1.000**	**1.000**	**1.000**	**0.825**
ImageNet ResNet-50	1.000	1.000	0.925	0.425
R3M ResNet-50	0.950	1.000	1.000	0.275
CLIP ResNet-50	1.000	1.000	0.975	0.625
ImageNet ViT-base	0.950	1.000	0.975	0.450
CLIP ViT-base	1.000	1.000	0.900	0.575
MAE ViT-base	1.000	1.000	0.925	0.350

TABLE I: Success rates of all visual representations trained with 100 demonstrations of indicated task variation. Non-pretrained ResNets clearly outperform pretrained models on ZR.

The performance on combinations of grasp variations (XTZR, ZTZR, and YRZR) is much worse when trained using the same amount of data (100 episodes). Success rates of the models range from **0.400** to **0.600** for Non-pretrained ResNet-18, the best performing model. We hypothesize that the amount of data (100 episodes) is not sufficient enough to capture the features to successfully complete the trajectory, and confirm that increasing the amount of demonstrations drastically improves performance from 0.38 average success rate to 0.65 and 0.70 average success rate (TABLE II).

	XTZR	ZTZR	YRZR	XZTYZR
Non-pretrained ResNet-18	**0.825**	**0.825**	**0.675**	**0.275**
Non-pretrained ResNet-50	0.425	0.775	0.300	0.075
ImageNet ResNet-50	0.225	0.225	0.175	0.050
R3M ResNet-50	0.150	0.275	0.05	0.050
CLIP ResNet-50	0.500	0.575	0.250	0.150
ImageNet ViT-base	0.150	0.300	0.225	0.025
CLIP ViT-base	0.300	0.250	0.200	0.050
MAE ViT-base	0.375	0.25	0.175	0.050

TABLE II: Success rates of all visual representations trained with 1000 demonstrations of indicated task variation using all objects.

Comparison of Task and Object Variations: We provide a more fine-grained evaluation of our best model (Non-pretrained ResNet-18) by running 3 different evaluation runs of 40 randomized episodes and reporting the mean and standard deviation over the 3 runs (TABLE III). "All" denotes the average of all evaluation runs above (total 360 episodes). These models are trained on 1000 demonstrations, which includes all shapes, with proprioception and 3 views (top + 2 wrist cameras).

We observe that the success rates are mostly consistent when evaluated on a completely new set of 40 randomized episodes, with a standard deviation falling around 0.05. We observe that while the success rate of rotation groups (Order-4: circle and plus, Order-2: minus, diamond, and hexagon, Order-1: u, pentagon, and arrow) decline consistently with increased degrees of symmetry. However, there is no clear trend in the order of performance for objects inside each group.

Objects	XTZR	ZTZR	YZR	XZTYZR
circle	0.85±0.07	1.00±0.00	0.83±0.05	0.43±0.07
plus	0.93±0.04	1.00±0.00	0.77±0.01	0.38±0.04
minus	0.80±0.03	0.98±0.00	0.44±0.10	0.33±0.10
diamond	0.77±0.06	1.00±0.00	0.33±0.08	0.34±0.08
hexagon	0.71±0.08	1.00±0.00	0.38±0.08	0.30±0.08
u	0.37±0.08	0.54±0.06	0.12±0.06	0.17±0.03
pentagon	0.34±0.10	0.56±0.04	0.10±0.07	0.18±0.07
arrow	0.38±0.07	0.66±0.08	0.18±0.03	0.17±0.05
key	0.38±0.07	0.66±0.08	0.17±0.04	0.19±0.05
all	0.61±0.02	0.82±0.03	0.37±0.05	0.28±0.03

TABLE III: Success rates of Non-pretrained ResNet-18 trained on 1000 demonstrations including all objects. Mean and standard deviations over 3 different evaluations of 40 randomized rollouts.

Comparison of Imitation Learning Methods: We compare our original setup, BC-MLP with various image encoders, against BC-RNN [1], a popular SOTA BC baseline that is shown to improve performance over BC-MLP. Instead of a MLP of layer sizes [1024, 1024, 18] on top of the image encoder, BC-RNN has two LSTM layers of hidden sizes [1024, 1024] and output size 18 that fuses image embeddings and proprioception from 10 most recent time frames as implemented in the original paper [1]. The tabulated results show success rates evaluated over 40 rollouts. These models are all trained on 1000 demonstrations with proprioception and 3 views (top + 2 wrist cameras).

Models	XTZR	ZTZR	YRZR	XZTYZR
ResNet-18 MLP	**0.825**	**0.825**	**0.675**	**0.275**
ResNet-18 RNN	0.525	0.300	0.350	0.075
CLIP ResNet-50 MLP	0.500	0.575	0.250	0.150
CLIP ResNet-50 RNN	0.300	0.325	0.150	0.200
CLIP ViT-base MLP	0.300	0.250	0.200	0.050
CLIP ViT-base RNN	0.150	0.350	0.025	0.025

TABLE IV: Success rates of Non-pretrained ResNet-18, CLIP ResNet-50, and CLIP ViT-Base with MLP vs LSTM action decoders, trained on 1000 demonstrations including all objects.

Contrary to our expectations, we observe that the performance is worse for BC-RNN on all tasks and models we ran these experiments on: Non-pretrained ResNet-18, CLIP ResNet-50, and CLIP ViT-base. We hypothesize that BC-RNN cannot use the history as well as it performed in Robomimic [18], because BC-RNN performs better on longer tasks while our tasks are relatively short. To be more specific, the 5 tasks in the RoboMimic have average episode lengths of 48 (Lift), 116 (Can), 151 (Square), 469 (Transport), and 480 (Tool Hang), where the performance of BC-RNN improved for only the longer three tasks. In comparison to their longer tasks, our tasks average around 10 to 40 frames. While ALOHA [28], another recent work also proposes improvements in temporal smoothing and consistency through use of transformers, we do not conduct experiment with their setup assuming insignificant improvements from the same reasons.

Comparison of Action Representation and Rotation

	Shape	XT	ZT	YR	ZR	XTZR	ZTZR	YRZR	XZTYZR
One model per shape	plus	0.9	1.0	0.7	1.0	0.6	0.7	0.2	0.3
	minus	1.0	1.0	1.0	0.7	0.4	0.5	0.3	0.0
	key	1.0	1.0	0.5	0.0	0.0	0.3	0.0	0.1
One model for all shapes	plus	0.3	0.2	0.2	0.9	0.2	0.5	0.1	0.0
	minus	0.5	0.2	0.1	0.5	0.2	0.6	0.0	0.0
	key	0.3	0.0	0.2	0.3	0.1	0.3	0.1	0.0

TABLE V: Real world results for one model per shape vs. one model for all shapes, trained from 10 demonstrations for each object

Rotation-Loss	XTZR	ZTZR	YRZR	XZTYZR
Quaternion - MSE	0.275	0.250	0.000	0.000
Quaternion - Frobenius	0.225	0.375	0.100	0.000
6D - Frobenius	0.575	0.625	0.200	0.125
6D - MSE	0.825	0.825	0.675	0.275

TABLE VI: Success rates of the Non-pretrained ResNet-18 using a combination of a certain rotation representation (either as a quaternion or as a 6d rotation matrix completed by Gram-Schmidt orthonormalization).

Regression Targets: The choice of action and rotation representations was crucial for learning the policy as described in TABLE VI. First, we observed that position control of the end-effector poses was the best action representation for the task, as velocity control was too sensitive to the step size of the action and was completely unable to correctly model trajectories that change sharply in the velocity space. Second, we observed that using 6D representations for 3D rotations proposed in [25], which regresses the first 2 columns of the rotation matrix and performs Gram-Schmidt orthonormalization to produce the full rotation matrix, greatly outperformed the standard quaternion representation.

C. Real World Results

We conduct real world experiments to verify if (1) our observations from the simulation experiments are consistent with the real world (2) our setup can generalize with fewer demonstrations on the real robot. We experiment with a setup that is more similar to our simulation setup by training a single model to generalize over all the shapes. We calculate the success rates by counting "smooth insertions," which does not include forcing the objects together such that objects slip into each other even when they are not aligned. In other words, we do not count unintended insertions that did not properly align the objects towards success. Additionally, we also experiment with a setup that has one model per shape.

We observe that one model for all shapes (TABLE V bottom) perform worse than models trained and evaluated in simulation, because of fewer number of demonstrations in the real world. In this setup, we were not able to draw conclusive observations on the exact trend of performance, due to high variance in success rates likely induced from limited number of train demonstrations (30 vs. 100/1000/10000) and evaluation runs (10 vs. 40). It is also worth noting that the real world setup does not have wrist cameras, adding to the performance drop compared to simulated experiments. In our additional experiments where we train one model per shape (TABLE V top), we notice considerable improvements in success rates, suggesting that more demonstrations are needed to generalize over multiple shapes. However, the general trend of performance is consistent with our simulation setup: (a) objects with more degrees of symmetry perform better (b) tasks with simpler variations perform better than combinations (e.g. ZR performs better than XTZR, ZTZR, YRZR, and XZTYZR for all shapes except the key).

D. Additional Details & Analysis

Apart from the core experiments mentioned above, we also provide additional details (Object Details, Training Hyperparameters) and experiments on the website.

Frozen vs. Finetuning: We conduct additional experiments by fine-tuning the pretrained image encoders with the action decoder to conclude that (a) Pretrained ResNet-50 models perform better when trained unfrozen, but still does not outperform from-scratch models and (b) Pretrained ViT-B/16 models are more unstable when trained unfrozen and often performs worse than the frozen counterpart.

Data Efficiency of Models: We compare the performane of all visual representations trained with 100, 1000, and 10000 training episodes to observe that the non-pretrained models consistently improves with more data while frozen models saturate in performance at 1000 demos due to the limited capacity of MLP action decoders.

VI. CONCLUSION

In this paper, we propose a novel dual-arm geometric-peg-in-hole task and evaluate the spatio-gemoetric reasoning capabilities of various visual representations through an imitation learning framework. We observe that a non-pretrained visual encoder with MLP policy decoder were most effective in completing the task across all variations. For future work, we plan to explore data-driven contact modelling by incorporating force feedback while also extending the geometric variation of objects to more realistic objects. Furthermore, we hypothesize that advanced imitation learning methods such as DiffusionPolicy [29], which is shown to perform well in multimodal learning tasks, may significantly improve performance on our task and leave it for future work.

ACKNOWLEDGMENT

We thank Bahaa Aldeeb and Alireza Rezazadeh for providing helpful feedback on our initial draft and all other members of the Robotics: Perception and Manipulation (RPM) Lab for their insightful discussions.

REFERENCES

[1] H. Park, J. Park, D.-H. Lee, J.-H. Park, M.-H. Baeg, and J.-H. Bae, "Compliance-based robotic peg-in-hole assembly strategy without force feedback," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 8, pp. 6299–6309, 2017.

[2] X. Li, R. Li, H. Qiao, C. Ma, and L. Li, "Human-inspired compliant strategy for peg-in-hole assembly using environmental constraint and coarse force information," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4743–4748.

[3] O. Azulay, M. Monastirsky, and A. Sintov, "Haptic-based and $se(3)$-aware object insertion using compliant hands," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 208–215, 2023.

[4] A. S. Morgan, Q. Bateux, M. Hao, and A. M. Dollar, "Towards generalized robot assembly through compliance-enabled contact formations," *arXiv preprint arXiv:2303.05565*, 2023.

[5] H.-C. Song, Y.-L. Kim, and J.-B. Song, "Guidance algorithm for complex-shape peg-in-hole strategy based on geometrical information and force control," *Advanced Robotics*, vol. 30, no. 8, pp. 552–563, 2016.

[6] W. Gao and R. Tedrake, "kpam 2.0: Feedback control for category-level robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2962–2969, 2021.

[7] B.-S. Lu, T.-I. Chen, H.-Y. Lee, and W. H. Hsu, "Cfvs: Coarse-to-fine visual servoing for 6-dof object-agnostic peg-in-hole assembly," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 402–12 408.

[8] K. Van Wyk, M. Culleton, J. Falco, and K. Kelly, "Comparative peg-in-hole testing of a force-based manipulation controlled robotic hand," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 542–549, 2018.

[9] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.

[10] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," *arXiv preprint arXiv:2203.12601*, 2022.

[11] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell, "Real-world robot learning with masked visual pre-training," in *2023 Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 205, 2023, pp. 416–426.

[12] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[13] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman, "Deep predictive policy training using reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2351–2358.

[14] Y. Duan, M. Andrychowicz, B. Stadie, O. Jonathan Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[15] D.-A. Huang, S. Nair, D. Xu, Y. Zhu, A. Garg, L. Fei-Fei, S. Savarese, and J. C. Niebles, "Neural task graphs: Generalizing to unseen tasks from a single video demonstration," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8565–8574.

[16] K. Zakka, A. Zeng, J. Lee, and S. Song, "Form2fit: Learning shape priors for generalizable assembly from disassembly," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9404–9410.

[17] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," in *2020 Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 155, 2021, pp. 726–747.

[18] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *arXiv preprint arXiv:2108.03298*, 2021.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *2021 International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 139, 2021, pp. 8748–8763.

[22] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 16 000–16 009.

[23] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, *et al.*, "Ego4d: Around the world in 3,000 hours of egocentric video," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 18 995–19 012.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[25] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5745–5753.

[26] Blender Foundation, "Blender, an open-source 3d computer graphics software." [Online]. Available: www.blender.org

[27] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6d object pose estimation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 16 611–16 621.

[28] T. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *2023 Robotics: Science and Systems (RSS)*, 2023.

[29] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *2023 Robotics: Science and Systems (RSS)*, 2023.