Learning Efficient and Robust Language-conditioned Manipulation using Textual-Visual Relevancy and Equivariant Language Mapping

Mingxi Jia[†1], Haojie Huang[†2], Zhewen Zhang[‡2], Chenghao Wang[‡2], Linfeng Zhao[2], Dian Wang[2], Jason Xinyu Liu[1], Robin Walters[2], Robert Platt[⋆2], Stefanie Tellex[⋆1]

†, ‡ Equal Contribution, ⋆ Equally Advising

[1]Brown University, [2]Northeastern University

***Abstract*— Controlling robots through natural language is pivotal for enhancing human-robot collaboration and synthesizing complex robot behaviors. Recent works that are trained on large robot datasets show impressive generalization abilities. However, such pretrained methods are (1) often fragile to unseen scenarios, and (2) expensive to adapt to new tasks. This paper introduces Grounded Equivariant Manipulation (GEM), a robust yet efficient approach that leverages pre-trained vision-language models with equivariant language mapping for language-conditioned manipulation tasks. Our experiments demonstrate GEM's high sample efficiency and generalization ability across diverse tasks in both simulation and the real world. GEM achieves similar or higher performance with orders of magnitude fewer robot data compared with major data-efficient baselines such as CLIPort and VIMA. Finally, our approach demonstrates greater robustness compared to large VLA model, e.g, OpenVLA, at correctly interpreting natural language commands on unseen objects and poses. Code, data, and training details are available `https://mingxi-jia.github.io/gem_page/`**

I. INTRODUCTION

Commanding a robotic manipulator with natural language is important for enabling human-robot collaboration, while adding language specifications increases the complexity of the policy learning problem because the model needs to deal with a higher dimensional state space, especially in the multi-task setting. Data-driven approaches open new possibilities for learning fine-grained language-conditioned policies by learning from human demonstrations. Yet, these methods require extensive data collection [1, 2, 3] or expensive finetuning [4, 5]. For example, CLIPort [1] needs 50-100 demonstrations per task to ensure robust performance. RT-2 [5] finetunes a 55B LLM model with 130k robot trajectories and 10 billion images-text pairs.

Another strand of prior works [6, 7, 8, 9] loosens the need for large robotic datasets by directly utilizing existing Vision Language Models (VLM) to generate robot commands. However, since the VLMs lack an understanding of the geometries of the physical world, these approaches are still fragile to unseen object poses and cannot perform accurate, fine-grained manipulation on novel shapes.

Therefore, two key weaknesses of prior work are (1) data-driven methods require extensive data collection and expensive fine-tuning if a new task is to be learned due to the complex state space of potential objects and poses as well as the large action space of possible robot movements, and (2) the zero-shot approches are fragile to unseen object poses and distractor objects. We address these challenges by introducing a new imitation learning algorithm that leverages textual-visual relevancies from pretrained VLMs and equivariant language mapping that enables our method to learn efficient yet robust language-conditioned policies.

Fig. 1: **Overview.** By leveraging textual-visual relevancy maps with equivariant language mapping, it allows the robot to learn robust manipulation policies from a few demonstrations with high data efficiency. Our method achieves, in average, 10x data efficiency against CLIPort [1] and 53% more robustness than OpenVLA [4].

Our approach, **G**rounded **E**quivariant **M**anipulation (**GEM**) exploits domain symmetries that exist in the language-conditioned manipulation problem, specifically, equivariance in $\mathrm{SE}(2)$. For example, consider "grasp the coffee mug by its handle." If there is a rotation or translation of the mug, the desired pick action should also transform accordingly, i.e., equivariantly. Our learned policy incorporates such language-conditioned symmetries and generalizes to unseen scenarios related by symmetric transformations, thus making the model generalize to unseen object poses in a few-shot manner. To enable novel object generalization across objects, colors, and shapes, we propose textual-visual relevancy maps to distill information from large vision and language models. We extract textual relevancy for open-vocabulary recognition and construct visual relevancy via

data retrieval to improve robustness. We leverage spatial action map [10], i.e., representing actions as pixel locations, with open-loop action primitives that allow us to flexibly leverage equivariance and semantic relevancies. The resulting method is robust to unseen object poses and achieves better novel object generalization ability, as shown in Figure 1.

We make the following specific contributions. **(1)** We propose a novel method to construct patch-level textual and visual relevancy maps for robust open-vocabulary generalization. **(2)** We analyze the symmetries underlying the language-conditioned manipulation problem and design a novel neural network called language steerable kernels for building language-conditioned equivariant networks. **(3)** We demonstrate the state-of-the-art generalization robustness and sample efficiency in both simulation and the real world on a series of challenging language-conditioned manipulation tasks, using only 10%-20% training data compared with CLIPort [1] and 0.1% training data compared with VIMA [2].

II. Related work

Language-conditioned Policy Learning: Using language as task specifications is a common way for multi-task policy learning. Prior works [11, 12, 13, 2, 14] use feature concatenation, FiLM [15], or the self/cross-attention mechanism to fuse image and language features. For example, Shridhar et al. [1] utilizes CLIP visual and text encoders and fuses language and image features with a two-stream architecture. Goyal et al. [14] processes the language token and visual token jointly with a transformer. These models treat language as static features, meaning that the language features remain the same no matter how visual features change in a specific task. These designs require lots of data to cover all possible variances of visual features to guarantee performance. For instance, VIMA [2] requires 60k robot demos to learn its visual pick & place tasks. In contrast, our method maps language instructions into equivariant features that dynamically apply over the entire image space to generate robot actions, allowing our method to achieve similar or higher success rates with only 10-20 demonstrations.

Data-efficient Few-shot Learning requires the robot to learn a robust policy to manipulate in-distribution objects given only a few demonstrations. Prior works using geometric-constrained models [10, 16, 17, 18, 19, 20, 21, 22, 23, 24] leverage symmetries in robotic tasks and have demonstrated their superior effectiveness for unseen pose generalization that allows few-shot learning. However, these methods often learn a single-task policy and only take visual inputs. It is challenging to incorporate language into equivariant models because, unlike images, language does not have explicit spatial properties like rotation or translation. In this paper, we propose a novel architecture that treats language as a dynamic feature and enables object-level equivariance using equivariant language mapping via Language Steerable Kernels. Our approach can learn a multi-task policy yet maintain high sample efficiency and spatial robustness.

Zero-shot Manipulation requires the robot to generalize to out-of-distribution objects with novel shapes, colors, or textures. Learning-from-scratch methods [10, 25, 26] perform well on seen objects but cannot generalize well on unseen ones. Prior work [27, 28, 29] uses LLM/VLMs as a zero-shot object detector or a text-level task planner, assuming access to skills like a grasping model and pose estimators. However, the lack of learning ability limits these methods from adapting to complex task-oriented behaviors, e.g., "insert the letter E block into the letter E hole" since there is no general actor available for such placing skills. F3RM [30] learns few-shot pick-and-place with distilled fields, but it is slow because it requires extensive camera views and test-time optimization for every action step. RT-2 [5] and OpenVLA [4] learn generalist vision-language-action models by training on large robotic datasets, which give a certain degree of emerging behaviors on unseen objects with novel instructions. Still, it is expensive to collect new datasets and finetune these large parameterized models to adapt to new tasks. In this paper, we propose a novel approach that is capable of learning policies in a predefined space of skills with a small number of demonstrations while leveraging the zero-shot generalization ability from pre-trained VLM models via patch-level textual and visual relevancy maps.

III. Method

Problem Statement and Assumptions: We consider the problem of learning from demonstrations for language-conditioned manipulation tasks in $\mathrm{SE}(2)$ space. We frame it as a two-step pick-and-place learning problem, although our approach can support a larger space of skills, including pushing, pulling, etc. Given a set of demonstrations that contains observation-language-action tuples (O_t, ℓ_t, a_t), the objective is to learn a policy $p(a_t|o_t, \ell_t)$ where the action $a_t = (a_t^{\text{pick}}, a_t^{\text{place}})$ has pick and place components. Please note the policy can be formulated to generate the multi-step pick-place actions, and we discard time step subscripts t for simplicity in the following sections. The visual observation O_t is a set of RGBD images at several camera views. The two-step actions, a^{pick} and a^{place}, are parameterized in terms of $\mathrm{SE}(2)$ coordinates $(u, v, \theta_{\text{pick}})$ and $(u, v, \theta_{\text{place}})$, respectively, where u, v denotes the pixel coordinates of the gripper position, θ_{pick} is the pick orientation defined with respect to the world frame, and θ_{place} is the delta angle between the pick and place. The language instruction ℓ_t specifies the current-step instruction, e.g., "open the drawer" or "grasp scissors by its handle and place into box." We assume ℓ_t for each step can be parsed into the pick instruction and the place instruction, $\ell = (\ell^{\text{pick}}, \ell^{\text{place}})$.

Method Overview: There are three main modules. **(1)** The relevancy extraction module takes multi-view images $\mathcal{O}$ and the language instruction ℓ. It outputs a dense relevancy map that summarizes the visual and language input. **(2)** The language-conditioned pick module takes as input the top-down orthographic RGB-D projection o of the scene with the language instruction ℓ and outputs an action map over pick actions. **(3)** Similarly, the language-conditioned place module produces an action map over place actions. The difference is

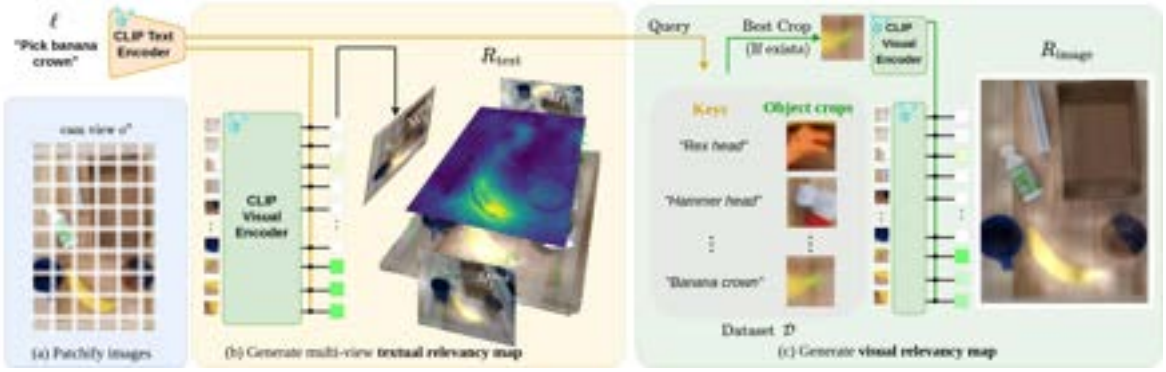


Fig. 2: **Relevancy Map Extraction.** After (a) **patchifying images** into patches, we compute two types of relevancy maps. (b) **Textual Relevancy Maps** allow open-vocabulary recognition, which is a topdown projection from the relevancy point cloud constructed with multi-view relevancy maps. We reconstruct the point cloud by reprojecting RGBD with camera intrinsics and merging different views with extrinsics. (c) **Visual Relevancy Maps** enhance the recognition robustness by querying from the existing database.

that the place module does convolution with an image-crop-conditioned kernel instead of a language-conditioned kernel.

A. Zero-shot Recognition via Textual & Visual Relevancies

The relevancy extraction module uses a pre-trained CLIP model to identify parts of the visual observation most relevant to the current task. Specifically, it takes the language goals ℓ^{pick} and ℓ^{place} and the current N-view observations $\mathcal{O} = \{o^1, o^2, ..., o^N\}$ as input and produces relevancy maps R^{pick} and R^{place} that highlight the language goals in the pixel space. Note that while these relevancy maps do not tell the system exactly where to pick, they provide a strong visual-language prior. The pipeline is illustrated in Figure 2.

Textual Relevancy Maps for Zero-shot Recognition: We use CLIP [31], which was trained by minimizing the cosine similarity between the image feature and its text label with internet data, to generate a pixel-wise relevancy score for each of the N views in $\mathcal{O}$. We split each image along a grid into image patches with patch size p and stride s. Each RGB image patch is then scored with its cosine similarity to the language instructions with pre-trained CLIP features. The textual relevancy function $\mathcal{R}_{text}$ can be described by

$$\mathcal{R}_{\text{text}}(\mathcal{P}(o^n), \ell) = \mathcal{P}^{-1}(\mathcal{E}_{\text{patch}} \cdot \mathcal{E}_\ell^T), \tag{1}$$

where $\mathcal{P}$ denotes the image patchification function and $\mathcal{P}^{-1}$ denotes an inverse process that transforms all similarity scores back to the original image dimension. $\mathcal{E}_{patch} \in \mathbb{R}^{(m \times n) \times d_m}$ is the embedding outputs from the CLIP image encoder, where $(m \times n)$ and d_m denote the number of image patches and the output embedding dimension of CLIP respectively. $\mathcal{E}_\ell \in \mathbb{R}^{1 \times d_m}$ is the embedding output for language instruction ℓ from the CLIP text encoder. After getting pixel-wise relevancies for each of the N views, we integrate this information into a single point cloud and label each point with the corresponding relevancy maps from all views so that we get a final top-down textual relevancy map R_{text} via projection, as shown in panel (b) of Figure 2.

Visual Relevancy Map for Semantic Correction: Although the textual relevancy map highlights image regions related to language instructions, we found it insufficient because the high-value region does not necessarily highlight the correct object if certain text is underrepresented during CLIP training. This misalignment creates noisy training samples as shown in Figure 3 where *Grounding DINO* only highlights a rough area and *Ours w/o Visual Relevancy* is biased by colors. To solve it, we further introduce the visual relevancy map, which is illustrated in Figure 2(c). Starting with the demonstrations, we identify the image crops in the demonstration data corresponding to pick and place events. For all pick/place events identified, we store into a database a pair comprised of the image patch at the pick/place location and the language query that describes the pick/place object (left side of Figure 2(c)). Then, at inference time, we index into the dataset using the language query text and recall the corresponding image crop, e.g., recall the image crop from the dataset corresponding to "banana root". The crop query process can be expressed by

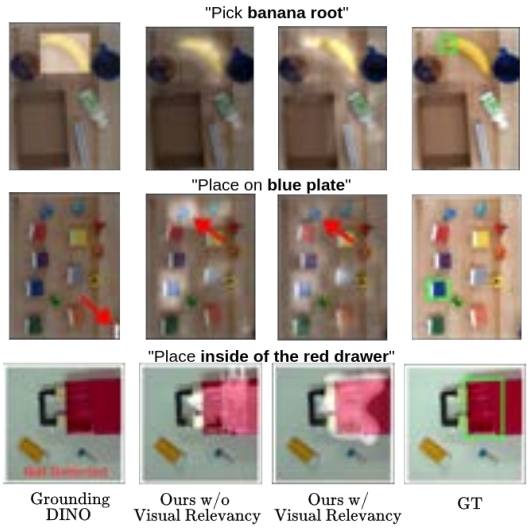


Fig. 3: **Qualitative Results of Relevancy Maps.** Our recognition accuracy with visual relevancy focuses more on the task-specific part (first-row), is less distracted by color (second-row), and recognizes uncommon instructions (third-row) than the baseline [32]. The text-visual ratio is set to 0.8 for visualization.

$$\text{QueriedCrop}(\mathcal{D}, \ell) = \underset{\text{crop} \in \mathcal{D}}{\arg\max}(\mathcal{E}_\ell \cdot K_{\text{crop}}^T), \tag{2}$$

where $K_{\text{crop}} \in \mathbb{R}^{N \times d_m}$ denotes all language embeddings

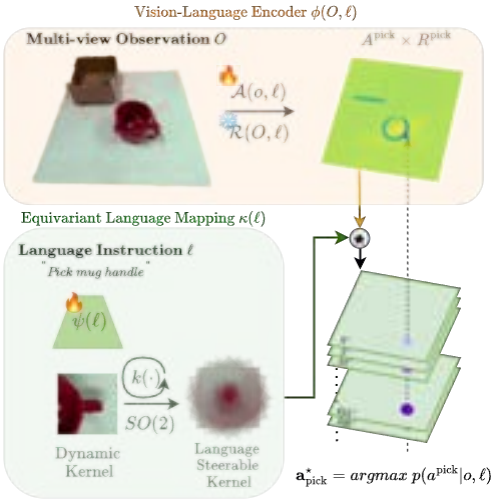


Fig. 4: **Object-level** $\mathrm{SE}(2)$**-equivariant Picking Model.** Our picking model f^{pick} consists of two branches. The top branch is our vision-language encoder $\phi(O, \ell)$, which includes a learnable action model $\mathcal{A}$ and the textual-visual relevancy extractor $\mathcal{R}$. The bottom branch is the Equivariant Language Mapping κ, which generates the language steerable kernel by rotating the dynamic kernel $\psi(\ell)$. The fire and the snowflake symbol denote frozen and trainable models. We discretize the $\mathrm{SO}(2)$ rotation space into 72 bins.

that correspond with N image patches for all N pick and place objects in dataset $\mathcal{D}$. $\mathcal{E}_\ell \in \mathbb{R}^{1\times d_m}$ denotes the embedding of the language query. Then, we generate visual relevancy maps by evaluating the cosine similarity between the CLIP embeddings of the recalled image crop and the patch embeddings from the top-down image o_t. The visual relevancy generation function $\mathcal{R}_{\text{visual}}$ can be described by

$$\mathcal{R}_{\text{visual}}(\mathcal{P}(o), \ell, \mathcal{D}) = \mathcal{P}^{-1}(\mathcal{E}_{\text{patch}} \cdot \mathcal{E}^T_{\text{crop}}), \tag{3}$$

where $\mathcal{E}_{\text{crop}} \in \mathbb{R}^{1\times d_m}$ denotes the embedding output for the image crop from the CLIP image encoder. If the pick/place target cannot be located in the dataset, i.e., $\max(\mathcal{E}_\ell \cdot K^T_{\text{crop}})$ is below a threshold, then the visual relevancy map function returns None. See Figure 3 for visualization examples.

To fuse textual and visual relevancy maps, we do a pixel-wise weighted averaging if a *Best Crop* exists in the dataset. Otherwise, only the textual map will be used.

B. Language Equivariance for Few-shot Learning:

While the relevancy maps provide semantic guidance for recognizing objects, the robot has no knowledge of where to manipulate the object shapes. Here, we use Learn from Demonstrations (LfD) to encode human knowledge into a learning-based model. A desired property we want a language-conditioned policy model to have is object-level equivariance, meaning that the output action should transform accordingly as the object specified by the language instruction transforms in the workspace. In the following, we propose a novel neural network architecture that enables equivariant language Mapping with its mathematical proofs.

Equivariant Language Mapping: We propose Equivariant Language Mapping via *language steerable kernels* that leverage object-level $\mathrm{SE}(2)$ equivariance, which allows our model to learn a new task in a few-shot learning manner. Previous work [10, 1] infers the pick location by taking the argmax on the output attention map from UNet, which has no guarantee that the model will generalize if any object moves in $\mathrm{SE}(2)$. Instead, we leverage the object-level symmetry by first mapping text embeddings by a UNet $\psi^{\text{pick}}(\ell^{\text{pick}})$ into a kernel with a spatial dimension of (C, H, W) and then converting it into steerable kernels by rotating the output with a group of n rotations $\{\frac{2\pi i}{n}|0 \le i < n\}$. This results in a stack of n rotated feature maps, $\Psi(\cdot) = \{g_0 \cdot \psi(\cdot), g_2 \cdot \psi(\cdot), \cdots, g_{n-1} \cdot \psi(\cdot)\}$, where $g_i = \frac{2\pi i}{n}$. We define the entire process as a language steerable kernel generator $\kappa^{\text{pick}}(\ell^{\text{pick}})$. The language kernel Φ maps language embeddings to convolutional kernels that satisfy the steerability constraints [33]. It allows picking action inference to be $\mathrm{SO}(2)$ equivariant with respect to the object poses.

Symmetry of Language-Conditioned Pick: The picking model f^{pick} calculates a probability distribution over gripper pose that corresponds to the probability of a successful grasp on the desired object part. This distribution $p(a^{\text{pick}}|o, \ell^{\text{pick}})$ is estimated by $f^{\text{pick}}(o, \ell^{\text{pick}}, M^{\text{pick}})$. The pick command executed by the robot is selected by $a^\star_{\text{pick}} = \arg\max a^{\text{pick}}$. The desired pick action is equivariant with respect to the pose of the object to be picked, i.e., $g \cdot p(a^{\text{pick}}|b^{\text{pick}}, \ell^{\text{pick}}) = p(a^{\text{pick}}|g \cdot b^{\text{pick}}, \ell^{\text{pick}})$, where b^{pick} denotes the object to be picked and $g\cdot$ denotes the action of a transformation g. Note that this equivariance is local to the object, in contrast to standard models that are equivariant w.r.t. the scene.

Specifically, assume the observation o_t contains a set of m objects $\{b_i\}_{i=1}^m$ on the workspace and denote the object b^{pick} as the goal object instructed by the language instruction ℓ^{pick}. If there is a transformation $g \in \mathrm{SE}(2)$ on the target object b^{pick} regardless of transformations on other objects, we denote it as $g \cdot o^{b^{\text{pick}}}$. The symmetry underlying f can be stated as

$$\begin{aligned} &\arg\max f^{\text{pick}}(g \cdot o^{b^{\text{pick}}}, \ell^{\text{pick}}) \\ &= g \cdot \arg\max f^{\text{pick}}(o, \ell^{\text{pick}}) \end{aligned} \tag{4}$$

Equation 4 claims that if there is transformation $g \in \mathrm{SE}(2)$ on the object b^ℓ, the best action $a^\star_{\text{pick}}$ to grasp the instructed object should be transformed to $g \cdot a^\star_{\text{pick}}$. If the symmetry is encoded in our pick model, it can generalize the knowledge learned from the demonstration to many unseen configurations. We use this symmetry to improve sample efficiency and the generalization of our pick model.

Pick Model Architecture: There are two main parts of the pick model. The first (shown in the top part of Figure 4) calculates a language-conditioned pick map as follows. We feed the raw RGB-D observation into a UNet, denoted as $\mathcal{A}^{\text{pick}}$, and encode ℓ^{pick}_t with the CLIP. The encoded language vector is concatenated onto the descriptor of every pixel in the bottleneck layer of $\mathcal{A}^{\text{pick}}$. The output of $\mathcal{A}^{\text{pick}}$ is denoted as $A^{\text{pick}}(o, \ell^{\text{pick}})$ or A^{pick} for simplicity. It is then integrated with the pick relevancy map R^{pick} with element-wise multiplication, shown as $\otimes$ in Figure 4.

Symmetry of Language-Conditioned Place: Place action that transforms pick target to the placement are bi-

equivariant [34, 35], i.e., independent transformations of the placement with g_1 and the pick target with g_2 result in a change ($a'_{\text{place}} = g_1 a_{\text{place}} g_2^{-1}$) to complete the rearrangement at the new configuration. Leveraging the bi-equivariant symmetries can generalize the place knowledge to different configurations and thus improve the sample efficiency. The coupled symmetries exist in the language-conditioned place:

$$\begin{aligned} \arg\max f^{\text{place}}(g_1 \cdot o^{b^{\text{place}}} + g_2 \cdot o^{b^{\text{pick}}}, \ell^{\text{place}}) = \\ g_1 \theta(g_2^{-1}) \cdot \arg\max f^{\text{place}}(o, \ell^{\text{place}}) \end{aligned} \tag{5}$$

where $g_1 \cdot o^{b^{\text{place}}} + g_2 \cdot o^{b^{\text{pick}}}$ denotes $g_1 \in \text{SE}(2)$ and $g_2 \in \text{SE}(2)$ acting on the instructed placement b^{place} and the picked object b^{pick}, respecitively. $\theta(g_2^{-1})$ denote the angle of the place action is rotated by $-g_2$. Specifically, the RHS of Equation 5 indicates that the best place location is rotated by g_1, and the place orientation is rotated by $\theta(g_1)\theta(g_2^{-1})$. Our place model is designed to satisfy the language-conditioned equivariance of Equation 5.

C. Equivariance Proof for Language Steerable Kernel

Here, we prove the object-level $\text{SE}(2)$ equivariance via language steerable kernels as stated in Equation 4.

Steeribility background: The G-steerable kernels are convolution kernels $K\colon \mathbb{R}^n \to \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ satisfying the *steerability constraint*, where n is the dimensionality of the space, d_{out} and d_{in} are the output and input field type

$$K(g \cdot x) = \rho_{\text{out}}(g) K(x) \rho_{\text{in}}(g)^{-1} \tag{6}$$

Proposition 1: If $\kappa(\ell)$ is a steerable kernel, it approximately satisfies the symmetry stated in Equation 4.

Translational Equivariance. Since FCNs are translationally equivariant by their nature, if a target object o^b is translated to a new location, the cross-correlation between $\kappa(\ell) * f^{\text{pick}}(o, \ell)$ will capture this translation, and there is no change in the change space.

Rotation Equivariance. Assuming ϕ satisfies the equivariant property that $\phi(T_g^0 o, \ell) = T_g^0 \phi(o, \ell)$ and the rotation of o^b is represented by $T_g^0 o_t$, we start the proof with lemma 1 and lemma 2.

Lemma 1: if $k(x)$ is a steerable kernel that takes trivial-type input signal, it satisfies $T_g^0 K(x) = \rho_{\text{out}}(g^{-1}) K(x)$.

Lemma 2: Cross-correlation satisfies that

$$(T_g^0 (K \star f))(\vec{v}) = ((T_g^0 K) \star (T_g^0 f))(\vec{v}) \tag{7}$$

Given Lemma 1 and lemma 2, we can prove that

$$\begin{aligned} \kappa(\ell) * \phi(T_g^0 o, \ell) =& \kappa(\ell) * T_g^0 \phi(o, \ell) \\ =& \kappa(\ell) * T_g^0 \phi(o, \ell) \\ =& T_g^0 T_{g^{-1}}^0 \kappa(\ell) * T_g^0 \phi(o, \ell) \\ =& T_g^0 [T_{g^{-1}}^0 \kappa(\ell) * \phi(o, \ell)] \text{ lemma 2} \\ =& T_g^0 [\rho_{\text{out}}(g) \kappa(\ell) * \phi(o, \ell)] \text{ lemma 1} \end{aligned}$$

It states that if there is a rotation on o, the grasp position is changed by T_g^0, and the rotation is changed by $\rho_{\text{out}}(g)$. Since the cross-correlation is calculated for each pixel without stride, the rotated b^ℓ is captured by $\rho(g)$. In our implementation, we generate the language-conditioned steerable kernel $\kappa(\ell)$ but remove the constraint of the equivariant property of ϕ. The U-Net encoder maintains soft global equivariance with the long skip connections and data augmentation.

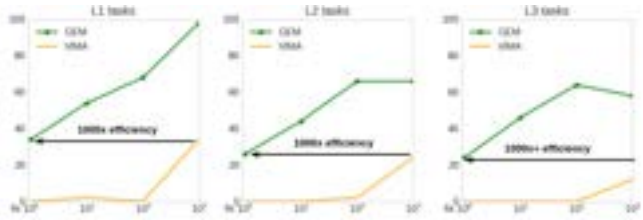


Fig. 5: **Performance Comparisons on VIMABench [2].** X-axis and y-axis represent the number of demonstrations for training and task success rate during evaluation in the *visual_manipulation* task.

IV. Experiments

We design the following experiments to answer four questions:

- How is our data efficiency and performance on seen and unseen scenarios compared with few-shot learning methods [1, 10], large transformer-based models [2], and zero-shot methods [6]? — Section IV-A
- How is the real-world performance of our method on different physical robot platforms? — Section IV-B
- How is our robustness against unseen poses and language & objects versus large VLA models [4] with few-shot fine-tuning? — Section IV-C
- How does each component in our method contribute to the final performance? — Section IV-D

A. Simulation Experiments

Tasks & Baselines For simulation tasks, we use 18 tasks provided by CLIPort Benchmark [1], including multi-step tasks (*block-in-bowl*, *packing-box/google-pairs/objects*), long-horizon (*stack-block-pyramid-seq*, *towers-of-hanoi*), deformable object (*align-rope*, *separating-piles*), novel color/object tasks (*unseen* tasks). The metric is in the range of 0 (failure) to 100 (success). Partial rewards are calculated in multi-step tasks. For instance, in *pushing-colored-piles*, pushing 10 piles out of 50 into the correct zone will be credited $\frac{10}{50} \times 100\%$ rewards. For baselines, we compare our method with four strong baselines: Transporter [10], CLIPort [1], VIMA [2], MOKA [6].

Comparing with Data-efficient CNN-based Methods: Transporter [10] and CLIPort [1] are data-efficient methods based on convolutional neural networks for pick-and-place tasks. Transporter-Lan is a modified Transporter [10] to take language instructions, where we concat the text embedding onto the bottleneck of UNet. In Table I, we report the performance on CLIPort Benchmark [1]. The best performance is highlighted in bold in each column. Several conclusions can be drawn from Table I. **(1)** GEM outperforms the baselines in all the tasks by a significantly large margin. For example, in task *separating-piles-unseen-colors*, our method gets 97.6% success rate with 20 demos while the best baseline only achieves 66.0%. **(2)** GEM is more sample efficient compared with the baseline. Trained with 10 demos, it can outperform

Model	packing-box-pairs seen-colors			packing-box-pairs unseen-colors			packing-seen-google objects-seq			packing-unseen-google objects-seq			packing-seen-google objects-group			packing-unseen-google objects-group		
	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100
Transporter-Lan [10]	50.4	72.4	86.8	41.2	40.4	60.7	36.3	57.0	83.2	31.7	46.8	54.9	49.6	57.0	81.2	56.6	59.4	77.4
CLIPort [1]	63.2	78.3	88.2	28.8	64.2	71.4	37.9	52.6	80.1	45.9	41.7	49.6	62.0	62.1	77.1	49.5	50.3	60.0
CLIPort-**multi** [1]	60.3	82.9	81.4	42.4	53.7	54.3	76.6	84.3	77.0	50.4	58.7	47.6	79.0	88.0	88.6	79.9	85.6	73.8
MOKA [6]	16.3			20.6			32.7			41.2			36.1			40.7		
GEM (ours)	79.6	86.7	91.8	67.3	71.4	**78.2**	76.2	85.8	89.7	69.2	**79.8**	**86.0**	86.6	85.1	**94.2**	78.1	71.9	82.3
GEM-**multi** (ours)	**90.6**	**90.7**	**93.8**	**73.8**	**78.2**	**78.2**	**93.7**	**91.0**	**90.3**	**86.3**	79.7	75.7	**94.5**	**93.1**	**94.2**	**89.7**	**90.9**	**88.5**

Model	stack-block-pyramid seq-seen-colors			stack-block-pyramid seq-unseen-colors			separating-piles seen-colors			separating-piles unseen-colors			towers-of-hanoi seq-seen-colors			towers-of-hanoi seq-unseen-colors		
	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100
Transporter-Lan [10]	52.0	72.7	94.3	18.0	26.0	17.0	40.0	60.0	92.0	56.0	73.8	52.3	81.1	88.6	95.7	43.4	48.3	60.0
CLIPort [1]	22.8	39.5	50.5	21.8	19.2	27.7	53.1	56.0	74.8	56.4	66.0	72.5	75.1	75.0	91.1	57.6	47.3	**99.4**
CLIPort-**multi** [1]	74.7	87.7	93.3	45.7	28.3	33.0	59.7	72.2	75.0	67.8	65.2	58.8	78.3	95.4	97.4	60.3	69.4	69.7
MOKA [6]	0			0			11.4			14.8			0			0		
GEM (ours)	70.7	82.7	**96.3**	59.3	73.7	**84.3**	82.3	75.4	78.8	60.0	91.8	96.6	88.3	93.4	**100**	83.1	87.7	98.0
GEM-**multi** (ours)	**94.3**	**95.3**	95.0	**76.0**	**89.3**	78.7	**94.2**	**96.2**	**92.0**	**89.0**	**97.6**	**96.6**	**96.3**	**99.4**	98.9	**93.4**	**98.0**	97.1

Model	align-rope			packing-unseen-shapes			assembling-kits-seq seen-colors			assembling-kits-seq unseen-colors			put-blocks-in-bowls seen-colors			put-blocks-in-bowls unseen-colors		
	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100	10	20	100
Transporter-Lan [10]	11.5	33.7	72.4	24.0	26.0	30.0	26.4	39.2	58.4	20.0	24.8	23.6	42.7	68.7	86.3	12.0	17.0	36.0
CLIPort [1]	30.0	16.9	51.5	29.0	24.0	34.0	17.8	24.8	39.4	16.6	20.6	36.6	37.2	55.6	92.7	50.8	41.7	51.8
CLIPort-**multi** [1]	39.7	42.4	40.8	52.0	46.0	52.0	28.8	42.8	32.0	28.4	27.2	18.8	84.0	96.0	98.0	38.7	48.0	44.0
MOKA [6]	2.4			4.0			0			0			22.5			17.5		
GEM (ours)	31.6	38.6	**69.0**	54.0	44.0	**52.0**	42.8	47.2	**62.4**	34.4	40.0	**62.8**	94.0	98.3	**100**	87.7	92.0	94.3
GEM-**multi** (ours)	**62.6**	**59.6**	58.6	**60.0**	**50.0**	**52.0**	**55.6**	**62.0**	56.8	**53.2**	**58.0**	46.4	**100**	**100**	**100**	**95.3**	**97.0**	**97.0**

TABLE I: **Performance Comparisons on CLIPort Benchmark Tasks (%)** on 50 testing episodes. {10, 20, 100} denotes the number of demonstrations used in training. "**-multi**" denotes multi-task models where they are trained on all tasks and evaluated separately.

the baselines with 20 and 100 demos on 10 out of 18 tasks. In *stack-block-pyramid-seq-seen-colors*, our method trained with 10 demos gets 70.7% while CLIPort only gets 50.5% trained with 100 demos. **(3)** GEM demonstrates strong zero-shot learning ability. The performance gap between GEM and the baselines becomes larger with unseen objects. In *put-blocks-in-bowls* with 100 demos, the difference between GEM and CLIPort increases from Δ7.3% to Δ42.5% on unseen colors. **(4)** GEM is capable of learning a multi-task policy from diverse datasets. As shown in multi-task results, GEM-multi performs best on 41 out of 54 evaluation cases.

Comparing with Zero-shot VLM Methods: In Table I, we also compare with MOKA [6], a state-of-the-art zero-shot method that is based on GPT4 [36] and SAM [37] without training on any robot data. Table I shows that our method trained with 10 demonstrations achieves, on average, 43.67 % better than the baseline across all tasks. The large performance gap indicates that (1) there is a large gap between LLM/VLM data and task-specific robot data, and (2) it is vital to do few-shot learning to align pre-trained relevancies and post-trained robotic features.

Comparing with Transformer-based Methods: VIMA [2] is a learning-based approach that is based on pretrained transformer architecture [38]. In Figure 5, we compare GEM with VIMA [2] on VIMABench [2], where ours achieves the same performance with 6 demos comparing VIMA with 6k demos.

B. Real-world Experiments

For real-world experiments, we evaluate the few-shot and zero-shot learning ability of our model on a series of challenging open-vocabulary manipulation tasks.

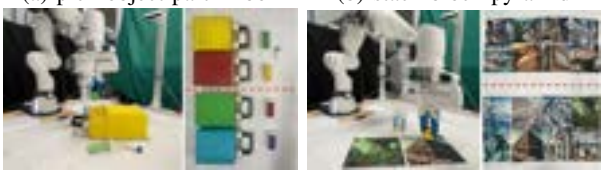
(a) UR5 (b) Franka (c) Spot

Fig. 6: **Real-world Tabletop and Mobile Manipulation Setup.** Cameras are highlighted in red, and workspaces are labeled in blue.

Tabletop Tasks: As shown in Figure 7, we design four distinguishable tasks to demonstrate the effectiveness of our method on different levels. *Pick-object-pat-in-box* demonstrates a key necessity of the few-shot learning ability of our method that we can learn to take fine-grained task-specific instructions like "pick red mug **by its handle**". *Stack-block-pyramid* shows the method's capability of doing long-horizon sequential tasks with an $\mathrm{SE}(2)$ model in 3D space with z-axis heuristics. *Pill-storaging* shows our method can do multi-task behaviors within one model beyond pick-and-place with simple primitive actions. *Common-knowledge* demonstrates zero-shot recognition ability on common knowledge objects.

Tabletop Results: We evaluate our method in four tasks as shown in Figure 7 and report the results in Table II. Object sets can be found in Figure 7. Our single-task method outperforms the baseline on all tasks up to a margin of 75.0%. On *pick-object-part-in-box*, our method reaches 76.7% success rate on seen objects while the baseline can only obtain 26.7% success rate. Besides, it shows strong generalization ability

(a) pick-object-part-in-box (b) stack-block-pyramid



(c) pill-organizing (d) common-knowledge

Fig. 7: **Real-world Tabletop Tasks.** The transparent and solid arms show the intermediate actions. On each task's right side, it shows the seen/unseen objects on the upper/bottom sides.

on unseen colors and shapes, whereas the baseline fails to generalize well. The real-world experiments further prove the few-shot and zero-shot learning ability of GEM.

	pick-object-part-in-box(5)	stack-block-pyramid(5)	pill-storaging(5)	common-knowledge(2)
CLIPort (seen)	8/30	46/60	8/48	5/16
GEM (seen)	**23/30**	**56/60**	**35/48**	**16/16**
CLIPort (unseen)	2/30	6/60	8/48	4/16
GEM (unseen)	**16/30**	**38/60**	**27/48**	**16/16**

TABLE II: **Real-world Tabletop Performance**. The numbers next to task names indicate the n demonstrations we collect per training object per task. "(seen)" denotes that all objects are in-distribution, but their poses are randomly initialized during testing. "(unseen)" means all testing objects are novel. Denominators are the number of evaluation steps. Evaluation length differs due to different task lengths and numbers of objects.

Mobile Manipulation Results: To demonstrate the effectiveness of our method in an open-world setting with a large workspace, we evaluate GEM with Spot on a language-conditioned pick-place task. Pixel-based action space allows us to collect all the demos on one table, and the policy can generalize to a multi-table environment by using an action parameterization trick, where we concat observations in spatial dimension and unionize the output action map during inference time. Our model reaches 80% success rate for seen objects and 50% for unseen objects.

C. Generalization Robustness Analysis:

We evaluate the generalization robustness in terms of three aspects: unseen translations, unseen rotations, and novel objects with unseen language instructions. We compare with OpenVLA [4], which is a large vision-language-action model pretrained on large-scale robotic datasets. We fine-tune OpenVLA on a few demonstrations and apply the same $\mathrm{SE}(2)$ data augmentation to guarantee fair comparison.

Spatial Generalization Robustness: We design a spatial generalization testbed to evaluate the translational and rotational generalization robustness. The task is to "pick the red mug by handle and place into brown box", where the robot needs to perform precise picking on the mug handle and put it into a box. We evenly collect demonstrations of 12

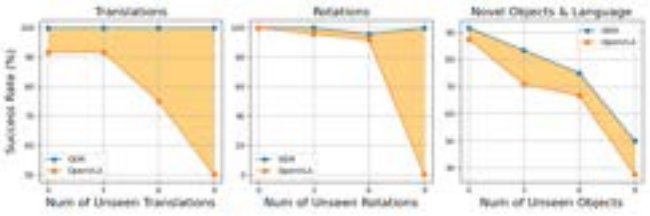


Fig. 8: **Robustness Analysis** on Positional and Object Generalization. We highlight our robustness gain over the baseline in orange.

translations and rotations across the workspace separately. To test the robustness of translation generalization, we gradually decreased the amount of training data so that the number of unseen poses increases during evaluation. As shown in Figure 8, our method preserves robust performance when testing on more unseen poses. Meanwhile, the baseline suffers a severe performance drop when the data is limited.

Novel Object Generalization Robustness: We construct an object set that consists of 12 objects. The task is to "pick {OBJECT} and place into brown box". To test the robustness of novel objects and language instruction, we gradually decrease the number of training objects n_{train} and test the models on the rest of the objects. During evaluation, the object orders are randomly shuffled so that all scenarios are unseen, even if they contain seen objects. OpenVLA [4] suffers more performance loss compared with our method when testing on more novel objects.

	pyramid-seen-100	pyramid-unseen-100
Ours	**94.0**	**84.3**
w.o Relevancy Map	91.7 (↓ 2.3)	21.0 (↓ 63.3)
w.o Multi-view Relevancy	63.0 (↓ 31.0)	34.3 (↓ 50.0)
w.o Visual Relevancy Map	91.3 (↓ 2.7)	82.7 (↓ 1.6)
w.o Steerable Kernel	7.7 (↓ 84.0)	1.7 (↓ 82.6)
w.o Lan-cond Kernel	90.7 (↓ 2.7)	61.3 (↓ 23.0)
CLIP → Ground DINO	83.7 (↓ 10.3)	21.7 (↓ 41.6)

TABLE III: **Ablation Study in Simulation.** Arrows indicate performance differences between ours and each variation.

	arrange-letter-to-word-seen	arrange-letter-to-word-unseen
Ours	**72.2**	**75.0**
w.o Visual Relevancy Map	42.2 (↓ 30)	68.8 (↓ 6.2)

TABLE IV: **Ablation on Visual Relevancy Map in Real-world Tasks.** 5 demos are used for training.

D. Ablation Study:

To investigate each component, we present a detailed ablation as shown in Table III and Table IV. The main findings are: (1) Relevancy maps are crucial for novel object generalization, where the performance for pyramid-unseen drops by 63.3 without it. (2) Language steerable kernels are important for few-shot learning from limited demonstrations. (3) Visual relevancy maps are necessary for real-world tasks where the performance drops 6.2%-30% without it. Since real-world observations are often noisier, visual relevancy maps help suppress the noise. (4) Interestingly, multi-view observations can significantly affect performance. We hypothesize that side-view observations are closer to natural image views on which CLIP is trained. (5) The performance drops if we replace CLIP with an open-vocabulary object detector Grounding DINO [32]. It shows that relevancy maps

are more scalable without the "objectness" inductive bias that object detectors often rely on.

	block-in-bowl-seen			block-in-bowl-unseen		
Number of Cameras	1	2	3	1	2	3
Ours	**98.0**	**100.0**	**100.0**	**79.0**	**88.3**	**96.0**
CLIPort	98.0	92.7	95.3	36.0	45.3	38.7

TABLE V: **Ablation on Number of Camera Views.** 100 demos are used for training. Evaluated at 20,000 Steps.

The performance of our model gets affected, but still remains good performance given fewer camera views. As shown in Table V, reducing camera views from three to one, the performance drops by 2.0 and 17.0 in *put-block-in-bowl-seen* and *put-block-in-bowl-unseen*, respectively. Our hypothesis is that: (1) increased occlusion issues arise from fewer viewpoints, and (2) noisier saliency maps result from the absence of smoothing effects by multiple views. In contrast, CLIPort maintains stable performance given any number of cameras. However, our model still outperforms the baseline in all testing cases. In addition, the choice of viewpoint in the single-view setting is important. We find that the front-view camera that looks at the workspace from the opposite side of the robot works better than other views.

V. Conclusion

In this work, we propose **G**rounded **E**quivariant **M**anipulation (**GEM**) that leverages pretrained textual-visual relevancy and the inherent symmetry in language-conditioned manipulation. Our method learns robust language-conditioned manipulation policies in a few-shot manner and shows a certain degree of novel object generalization ability. We validate the proposed method on simulated and real-world tasks against different baselines. One limitation is that our model needs predefined action primitives with heuristics like "pick", "place", "open", etc, limiting its ability to perform more complex tasks. Fortunately, LLMs [36] and LLM-based methods [39] provide convenient tools to parse and translate natural languages into high-level robot primitives. In the future, we will extend our method to (1) $\mathrm{SE}(3)$ action space without predefined primitives, and (2) improve local understanding [40, 41]. Worth mentioning, the language-conditioned symmetries we studied in Section III are applicable for $\mathrm{SE}(3)$ space, which provides a solid foundation for extension using 3D convolution methods [42, 34].

Acknowledgment

This work is supported by ONR under grant numbers N000142412784.

References

[1] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *CoRL*, 2022, pp. 894–906.
[2] Y. Jiang *et al.*, "Vima: General robot manipulation with multimodal prompts," *arXiv:2210.03094*, 2022.
[3] A. Brohan *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv:2212.06817*, 2022.
[4] M. J. Kim *et al.*, "Openvla: An open-source vision-language-action model," *arXiv:2406.09246*, 2024.
[5] A. Brohan *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv:2307.15818*, 2023.
[6] F. Liu *et al.*, "Moka: Open-vocabulary robotic manipulation through mark-based visual prompting," *arXiv:2403.03174*, 2024.
[7] W. Huang *et al.*, "Voxposer: Composable 3d value maps for robotic manipulation with language models," in *CoRL*, 2023, pp. 540–562.
[8] S. Mirchandani *et al.*, "Large language models as general pattern machines," *arXiv:2307.04721*, 2023.
[9] J. Liang *et al.*, "Code as policies: Language model programs for embodied control," in *ICRA*, 2023, pp. 9493–9500.
[10] A. Zeng *et al.*, "Transporter networks: Rearranging the visual world for robotic manipulation," in *5th CoRL*, 2021.
[11] L. Shao *et al.*, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," *IJRR*, vol. 40, 2021.
[12] G. Tziafas *et al.*, "Language-guided robot grasping: Clip-based referring grasp synthesis in clutter," *arXiv:2311.05779*, 2023.
[13] C. Tang *et al.*, "Task-oriented grasp prediction with visual-language inputs," *arXiv:2302.14355*, 2023.
[14] A. Goyal *et al.*, "Rvt: Robotic view transformer for 3d object manipulation," *arXiv:2306.14896*, 2023.
[15] E. Perez *et al.*, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, vol. 32, no. 1, 2018.
[16] H. Huang *et al.*, "Equivariant Transporter Network," in *RSS*, New York City, NY, USA, June 2022.
[17] D. Wang, R. Walters, and R. Platt, "SO (2)-equivariant reinforcement learning," in *ICLR*, 2022.
[18] H. Huang *et al.*, "Edge grasp network: A graph-based se (3)-invariant approach to grasp detection," in *ICRA*. IEEE, 2023, pp. 3882–3888.
[19] B. Hu *et al.*, "Orbitgrasp: Se (3)-equivariant grasp learning," in *8th CoRL*, 2024.
[20] L. Zhao *et al.*, "E(2)-equivariant graph planning for navigation," *IEEE Robotics and Automation Letters (RA-L)*, 2024.
[21] A. Simeonov *et al.*, "Neural descriptor fields: Se (3)-equivariant object representations for manipulation," in *ICRA*, 2022, pp. 6394–6400.
[22] J. Yang *et al.*, "Equivact: Sim (3)-equivariant visuomotor policies beyond rigid object manipulation," *arXiv:2310.16050*, 2023.
[23] L. Zhao *et al.*, "Integrating symmetry into differentiable planning with steerable convolutions," in *ICLR*, 2023.
[24] D. Wang *et al.*, "Equivariant diffusion policy," in *8th Annual Conference on Robot Learning*, 2025.
[25] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.
[26] C. Wang *et al.*, "Mimicplay: Long-horizon imitation learning by watching human play," in *7th CoRL*, 2023.
[27] A. Rashid *et al.*, "Language embedded radiance fields for zero-shot task-oriented grasping," in *7th CoRL*, 2023.
[28] M. Ahn *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv:2204.01691*, 2022.
[29] Y. Hu *et al.*, "Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning," *arXiv:2311.17842*, 2023.
[30] W. Shen *et al.*, "Distilled feature fields enable few-shot language-guided manipulation," *arXiv:2308.07931*, 2023.
[31] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021, pp. 8748–8763.
[32] S. Liu *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv:2303.05499*, 2023.
[33] G. Cesa, L. Lang, and M. Weiler, "A program to build e (n)-equivariant steerable cnns," in *ICLR*, 2021.
[34] H. Huang *et al.*, "Fourier transporter: Bi-equivariant robotic manipulation in 3d," *arXiv:2401.12046*, 2024.
[35] H. Ryu *et al.*, "Equivariant descriptor fields: Se (3)-equivariant energy-based models for end-to-end visual robotic manipulation learning," *arXiv:2206.08321*, 2022.
[36] OpenAI, "Gpt-4 technical report," 2023.
[37] A. Kirillov *et al.*, "Segment anything," in *ICCV*, 2023, pp. 4015–4026.
[38] A. Vaswani *et al.*, "Attention is all you need," in *NeurIPS 2017*, I. Guyon *et al.*, Eds., 2017, pp. 5998–6008.
[39] Y. Li *et al.*, "Hamster: Hierarchical action models for open-world robot manipulation," *arXiv preprint arXiv:2502.05485*, 2025.
[40] S. Rawlekar *et al.*, "Disentangling clip features for enhanced localized understanding," *arXiv preprint arXiv:2502.02977*, 2025.
[41] D. Jing *et al.*, "Fineclip: Self-distilled region-based clip for better fine-grained understanding," *Advances in Neural Information Processing Systems*, vol. 37, pp. 27 896–27 918, 2024.
[42] S. James *et al.*, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," in *CVPR*, 2022.