CodeDiffuser: Attention-Enhanced Diffusion Policy via VLM-Generated Code for Instruction Ambiguity

Guang Yin[3*] Yitong Li[4*] Yixuan Wang[1*] Dale McConachie[2] Paarth Shah[2]
Kunimatsu Hashimoto[2] Huan Zhang[3] Katherine Liu[2] Yunzhu Li[1]
[1]Columbia University [2]Toyota Research Institute [3]University of Illinois Urbana-Champaign [4]Tsinghua University
https://robopil.github.io/code-diffuser/

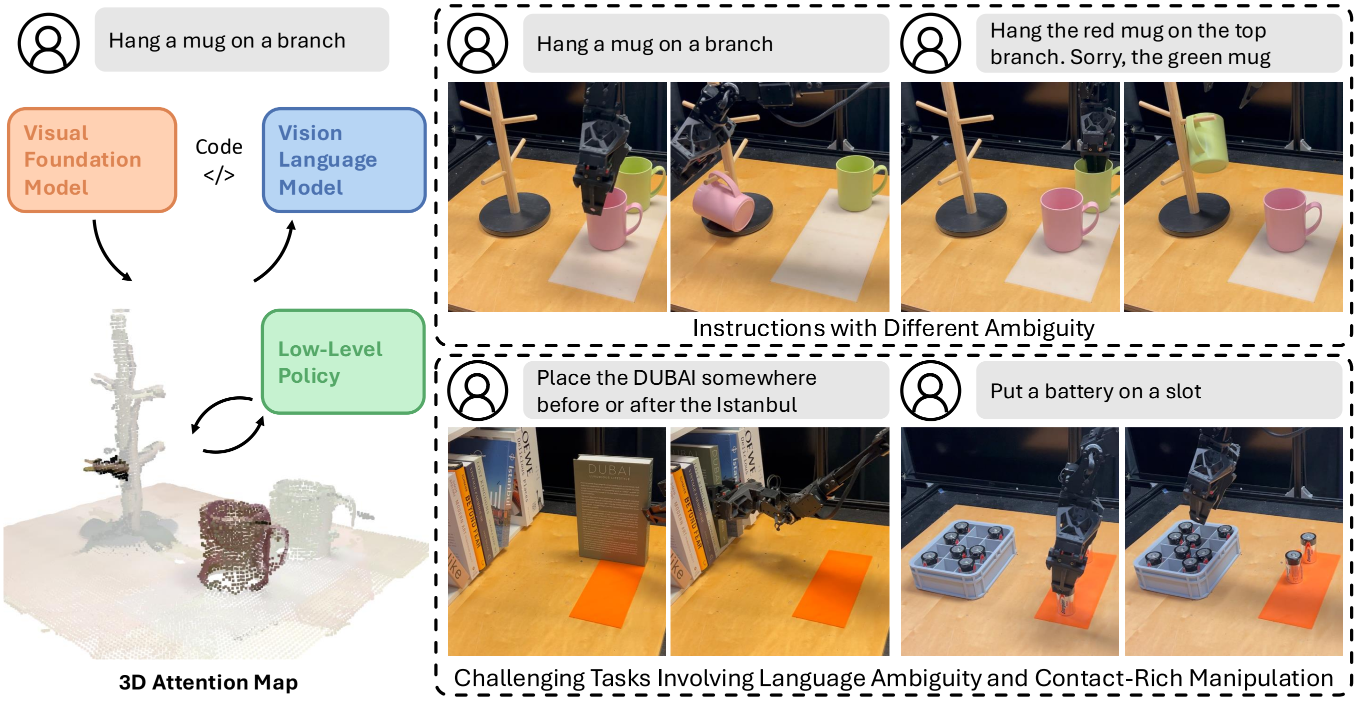

Hang a mug on a branch
Visual Foundation Model
Code </>
Vision Language Model
Low-Level Policy
3D Attention Map
Hang a mug on a branch
Hang the red mug on the top branch. Sorry, the green mug
Instructions with Different Ambiguity
Place the DUBAI somewhere before or after the Istanbul
Put a battery on a slot
Challenging Tasks Involving Language Ambiguity and Contact-Rich Manipulation

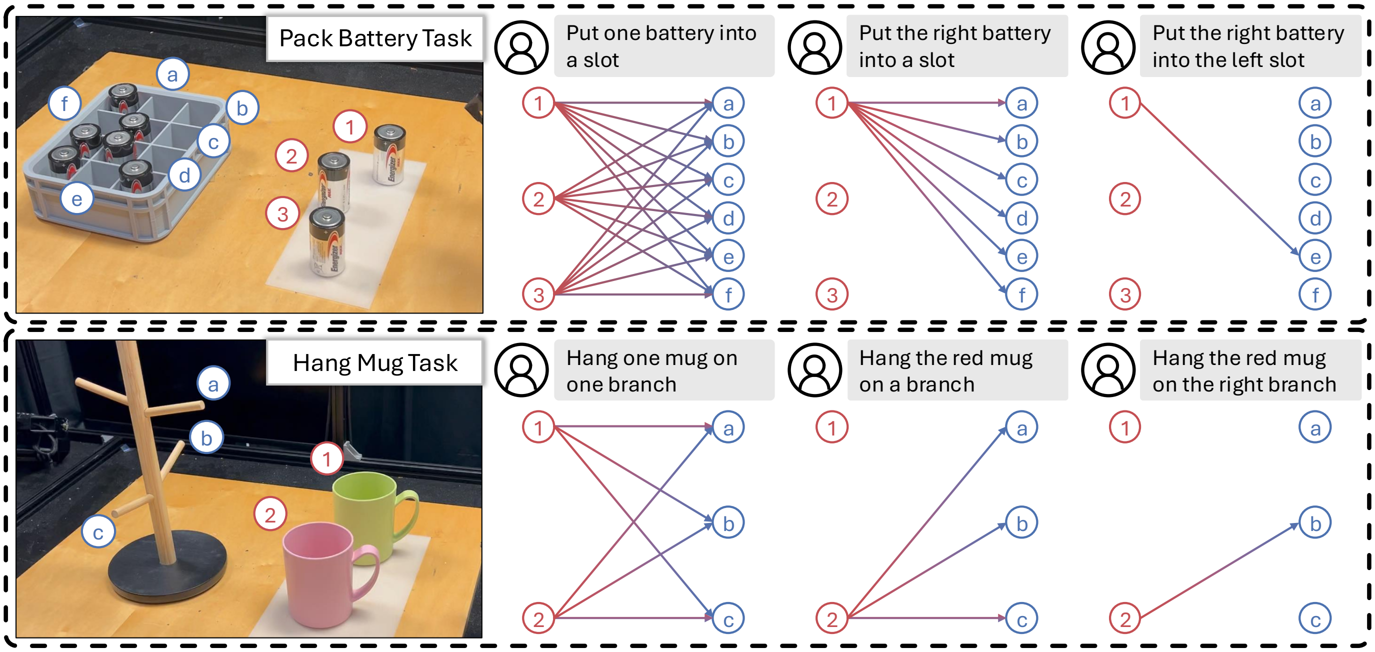
Fig. 1: **CodeDiffuser** leverages the code generated by Vision-Language Models (VLMs) as an interpretable and executable representation to understand abstract and ambiguous language instructions. This generated code interfaces with Visual Foundation Models (VFMs) to compute 3D attention maps, serving as an intermediate representation that highlights task-relevant areas and communicates with the low-level policy. Through extensive evaluations in both simulation and real-world settings, we demonstrate our method's effectiveness in challenging language-conditioned robotic tasks involving language ambiguity, contact-rich manipulation, and multi-object interactions.

Abstract—**Natural language instructions for robotic manipulation tasks often exhibit ambiguity and vagueness. For instance, the instruction "Hang a mug on the mug tree" may involve multiple valid actions if there are several mugs and branches to choose from. Existing language-conditioned policies typically rely on end-to-end models that jointly handle high-level semantic understanding and low-level action generation, which can result in suboptimal performance due to their lack of modularity and interpretability. To address these challenges, we introduce a novel robotic manipulation framework that can accomplish tasks specified by potentially ambiguous natural language. This framework employs a Vision-Language Model (VLM) to interpret abstract concepts in natural language instructions and generates task-specific code — an interpretable and executable intermediate representation. The generated code interfaces with the perception module to produce 3D attention maps that highlight task-relevant regions by integrating spatial and semantic information, effectively resolving ambiguities in instructions. Through extensive experiments, we identify key limitations of current imitation learning methods, such as poor adaptation to language and environmental variations. We show that our approach excels across challenging manipulation tasks involving language ambiguity, contact-rich manipulation, and multi-object interactions.**

I. Introduction

Natural language instructions are often vague and ambiguous when used to specify robotic tasks. As shown in Figure 2, the instruction "Pack the battery into the tray" could involve multiple feasible execution paths. However, current language-conditioned imitation learning methods typically deploy end-to-end models to jointly handle high-level semantic under-

[*] Denotes equal contribution.



Pack Battery Task
a
b
c
d
e
f
1
2
3
Put one battery into a slot
Put the right battery into a slot
Put the right battery into the left slot
Hang Mug Task
a
b
c
1
2
Hang one mug on one branch
Hang the red mug on a branch
Hang the red mug on the right branch

Fig. 2: **Language Ambiguity for Task Specification.** Natural language instructions for robotic tasks often exhibit inherent ambiguity and vagueness. Consider the instruction "Put one battery into a slot"–a common task in factory settings. In the given scenario, this instruction can be executed through multiple possible actions; the robot can choose from three available batteries and place one into any of the six potential slots in the tray, resulting in a total of eighteen possible choices. Furthermore, language instructions can vary in ambiguity, ranging from highly ambiguous directives to those that explicitly specify the target instance and goal location.

standing and low-level action prediction, which can lead to suboptimal performance. In our experiments, we found that existing diffusion policies converge to a notably low success rate—well below practical usability—on challenging tasks involving language ambiguity, even with extensive amounts of data [1, 2].

In this work, we propose a novel robotic manipulation framework to accomplish tasks with potential language ambiguities. Building upon the key technical insight that code generation provides an interpretable and executable interface between the visual-semantic understanding of Vision-Language Models (VLMs) and the dexterous physical capabilities of visuomotor policies, we introduce CodeDiffuser in this work to effectively handle language ambiguity and enhance semantic understanding capabilities. Our system consists of three major components: code generation using VLMs, 3D attention computing using Visual Foundation Models (VFMs), and low-level visuomotor policy. The VLM first processes both visual observations and language descriptions and uses the provided perception API to generate code that produces a 3D attention map, highlighting task-relevant areas by leveraging visual features from foundational vision models such as DINOv2 and SAM [3, 4]. The 3D point cloud and attention map are then fed into the visuomotor policy, which generates an action trajectory to complete the task.

We conduct comprehensive experiments to analyze the limitations of state-of-the-art imitation learning algorithms when handling language ambiguities. Our results show that performance declines as language ambiguity increases. Furthermore, we demonstrate that simply increasing the number of demonstrations does not significantly enhance the ability to address language ambiguity and vagueness.

To gain deeper insights into attention map conditioning, we evaluate the language-conditioned 3D attention map and the visuomotor policy separately. Both quantitative and qualitative results show that 3D attention maps generated by VLM-based code accurately align with human instructions and effectively highlight task-relevant locations. Furthermore, extensive quantitative analysis demonstrates that the 3D attention map serves as an effective representation, improving the visuomotor policy's ability to handle language ambiguity. Finally, we evaluate the entire system and show that CodeDiffuser successfully performs challenging robotic manipulation tasks with language ambiguity and vagueness.

In summary, our contributions are threefold: 1) We systematically evaluate the key limitations of state-of-the-art imitation learning frameworks in scenarios with language ambiguity and vagueness, such as pick-and-place tasks with multiple possible objects and destinations. 2) We propose CodeDiffuser, a novel robotic manipulation framework that addresses these challenges using VLM-generated code as an interpretable and executable intermediate representation. By interfacing with perception APIs, it generates 3D attention maps to bridge visual-semantic reasoning and low-level trajectory prediction. 3) We conduct extensive evaluations of individual modules and the full system in both simulation and real-world tasks, including contact-rich 6-DoF manipulation with multi-object interactions, demonstrating the effectiveness of our approach in handling language ambiguity.

II. Related Works

A. Imitation Learning for Robotic Manipulation

Imitation learning has achieved impressive results in dexterous robotic manipulation tasks, such as spreading sauces, flipping mugs [1, 2], tying shoelaces [5], and more [6–35]. While existing imitation learning frameworks model policies in an end-to-end manner—requiring them to jointly understand high-level semantics and predict low-level skills—this often results in suboptimal performance in complex scenarios where instructions lack specificity. In contrast, our framework leverages the visual-semantic reasoning capabilities of VLMs to interpret potentially vague or abstract natural language instructions and generate code—an interpretable and executable intermediate representation—that interfaces with perception APIs to provide structured inputs to the low-level policy.

B. Foundational Vision Model for Robotics

Foundational vision models have demonstrated impressive zero-shot generalization capabilities in 2D vision tasks such as detection, segmentation, and visual representation [3, 36–38]. Many robotic systems have adopted these models due to their strong generalization ability [20, 21, 35, 39–58]. Among these, GenDP and 3D Diffuser Actor are most relevant to our work [40, 45]. While GenDP demonstrates category-level generalization, it lacks the ability to understand natural language instructions. In contrast, our framework is capable of understanding potentially ambiguous natural language instructions by using visual-semantic reasoning capabilities of VLM and generated code as an intermediate representation.

The 3D Diffuser Actor introduces a diffusion model that predicts end-effector keyposes from point clouds augmented with visual features from foundational vision models. In contrast, our approach uses VLM-generated code to compute 3D attention map, which highlights task-relevant regions and possesses much lower dimension compared to 3D Diffuser Actor for easier visuo-motor policy training. Furthermore, 3D Diffuser Actor predicts end-effector keyposes rather than a continuous trajectory, which requires manual keypose definition for each task and is insufficient for tasks requiring smooth trajectory control, such as stowing books. In contrast, our low-level visuomotor policy predicts a smooth trajectory, providing greater flexibility for diverse tasks.

C. Code Generation for Robotics

Due to their advanced visual-semantic reasoning and code generation capabilities, LLMs and VLMs have been widely applied to generate code for various robotic tasks, including manipulation [41, 42, 59–73], navigation [58, 74], and planning [75, 76], as reviewed in several surveys [77–80]. Existing works typically employ motion planning to generate trajectories [41, 42]. While these methods demonstrate impressive zero-shot performance, they lack the ability to learn skills from human demonstrations. In contrast, our framework connects a VLM to the low-level visuomotor policy via VLM-generated code, seamlessly combining the high-level semantic reasoning capabilities of VLM with the smooth low-level control enabled by learned policies.

Among LLM- and VLM-based approaches applied to robotics, VoxPoser and ReKep [41, 42] are closely related to our work. They use a 3D voxel heatmap and 3D relational keypoints, respectively, as intermediate representations to encode both geometric and semantic information, incorporating various optimization methods to generate action trajectories for downstream manipulation tasks. However, for certain tasks—such as stowing books, which involve multi-object interactions—instantiating an optimization problem can be challenging due to the complex dynamics involved. In contrast, our approach learns a visuomotor policy from demonstrations, offering greater flexibility in handling a wide range of tasks.

III. Method

A. Problem Statement

Our learning goal is to match the distribution given in a dataset of collected demonstrations, i.e.,

$$\min_{\pi} L(p_{\text{pred}}(a|o_t, l), p_{\text{gt}}(a|o_t, l)), \tag{1}$$

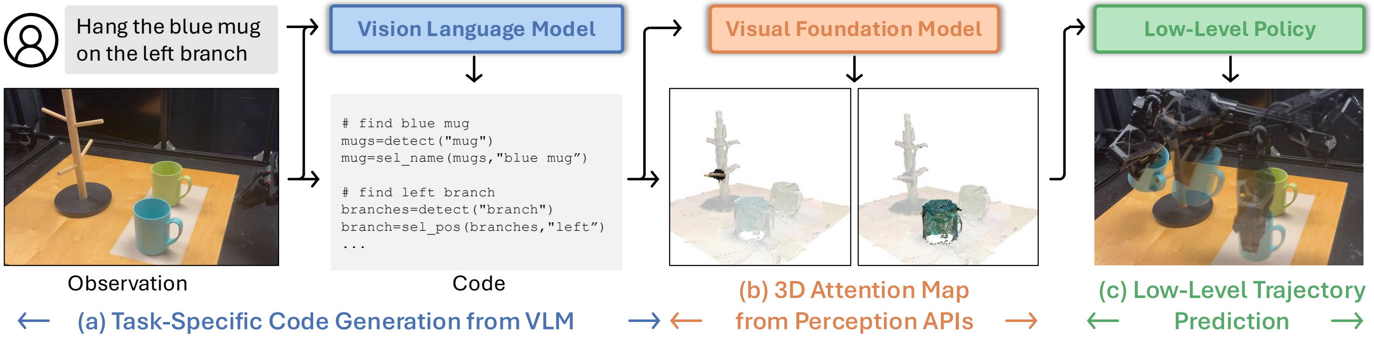
with observations $o \in \mathcal{O}$, task descriptions $l \in \mathcal{L}$, and actions $a \in \mathcal{A}$. $p_{\text{pred}}(a|o_t, l)$ represents the predicted action distribution and $p_{\text{gt}}(a|o_t, l)$ denotes the ground truth action distribution provided by a human demonstration dataset. The action distribution $p(a|o_t, l)$ can be decomposed as follows:

$$\begin{aligned} p(a|o_t, l) &= \int_{z_t} p(a|o_t, l, z = z_t)p(z = z_t|o_t, l) && (2) \\ &= \int_{z_t} p(a|z = z_t)p(z = z_t|o_t, l), && (3) \end{aligned}$$

where z_t is a task-relevant latent representation of the state such that $p(a|o_t, l, z = z_t) = p(a|z = z_t)$, i.e., z_t contains enough information about the observation and instruction to predict the action.

We observe that in the presence of language ambiguity, the distribution $p(a|o_t, l)$ is highly multi-modal. While single-task imitation learning policies have recently shown great success in modeling the multi-modal action distributions [2], ambiguity in the instruction introduces an additional complexity. For instance, in the packing battery task illustrated in Figure 2, if the instruction is “Hang one mug on one branch” without specifying the mug or branch instance, the probability of each battery-slot pair is $1/18$, imposing an additional axis of multi-modality in the language instructions. The highly multimodal distribution poses challenges for training an end-to-end policy to model $p(a|o_t, l)$. In practice, we find that learning a single end-to-end policy is suboptimal when dealing with task ambiguity. Notably, we show in Section IV-B that the current state-of-the-art methods can fail to achieve a high success rate even with extensive training demonstrations.

Under this framework, we model $p(a|z = z_t)$ with modern generative imitation learning methods and $p(z = z_t|o_t, l)$ with a VLM. We adopt a latent representation indicating task-relevant areas to accomplish the task, i.e., a 3D attention


Hang the blue mug on the left branch
Vision Language Model
Visual Foundation Model
Low-Level Policy
Observation
Code
(a) Task-Specific Code Generation from VLM
(b) 3D Attention Map from Perception APIs
(c) Low-Level Trajectory Prediction

```
# find blue mug
mugs=detect("mug")
mug=sel_name(mugs,"blue mug”)

# find left branch
branches=detect("branch")
branch=sel_pos(branches,"left”)
...
```

Fig. 3: **Method Overview.** CodeDiffuser consists of three primary components: code generation, 3D attention map computation, and low-level policy. (a) CodeDiffuser first leverages the VLM semantic reasoning and code generation capabilities to understand human instructions with potentially ambiguity and environmental observations, generating task-specific code. (b) This generated code interfaces with perception APIs, built upon the VFM, to compute 3D attention maps that highlight task-relevant areas. (c) The attention maps are then fed into a low-level policy to generate actions accomplishing tasks with multi-object interaction, contact-rich manipulation, and language ambiguity.

map highlighting a specific mug-branch pair. We show that this representation can be used to condition the imitation learning algorithms, and can also be generated from language by leveraging the reasoning capabilities of VLMs. We first generate intermediate code from the instruction l and multi-view RGBD observations $o_t \in \mathbb{R}^{K\times H\times W\times 4}$, where K is the number of camera views, and H and W represent the image height and width, respectively, as described in Section III-B. In Section III-C, we describe the API provided to the code generation process used to construct our state representation z_t, a 3D attention map that highlights task-relevant regions. Finally, this 3D attention map is input into the low-level policy, which predicts a sequence of 6D end-effector poses a for the robot, as illustrated in Section III-D.

B. Code Generation

CodeDiffuser uses a VLM to translate a natural language instruction into executable code that generates 3D attention maps for downstream consumption by the visuomotor policy. In our work, we leverage the VLM's few-shot generalization and commonsense reasoning capabilities to generate perception code from a few provided examples. We first provide a VLM, such as ChatGPT-4o [81], with several in-context example code snippets for constructing 3D attention maps using our perception APIs. At inference time, we input a task description into the VLM, which then produces the code to create the 3D attention map. The generated code takes as input the RGB observations and outputs a 3D attention map. Importantly, the code generation process can generate code to further invoke VLMs, enabling more advanced semantic reasoning. For instance, as shown in Figure 3 (a), if the instruction is "Hang the blue mug on the left branch," the generated code first detects all instances of mugs. It then selects the "blue mug" instance from the detected mugs using another call to the VLM. A similar process is applied for branches, with the key difference being that instance selection for branches relies more on spatial relations, such as "left" rather than semantic attributes like "blue". The implementation of perception APIs is detailed in Section III-C.

C. API for 3D Attention Map Generation

To provide the VLM with an API to bridge the semantic meaning of an instruction and the corresponding 3D observation, we design a simple yet effective API, including the functions `detect`, `sel_name`, and `sel_pos`. Example usage of these functions is provided in-context during inference.

`detect`: The `detect` function takes multi-view RGBD observations from cameras with known intrinsics and extrinsics and an object name as input and outputs a list of object instances belonging to the object category. The detection pipeline includes 3D feature extraction and object clustering. We first extract 2D feature maps $\mathcal{W}_i \in \mathbb{R}^{H\times W\times F}$ of each view using the DINOv2 model [3], where F is the semantic feature dimension. Utilizing the depth images and camera parameters from all K viewpoints, we fuse the 2D DINOv2 features into 3D space to obtain 3D point clouds $\mathcal{P} \in \mathbb{R}^{M\times 3}$ and associated 3D features $\mathcal{F} \in \mathbb{R}^{M\times F}$, where M is the number of points. In practice, we follow the implementation introduced in D^{3}Fields [39] to obtain the 3D point clouds and their associated semantic features.

From a reference object, we annotate a set of relevant images, obtaining a reference DINOv2 feature denoted as $\mathcal{F}_{\text{ref}} \in \mathbb{R}^F$. We observe that the annotation process is lightweight and is only required when first setting up a task for training. By comparing reference feature $\mathcal{F}_{\text{ref}}$ with 3D semantic features $\mathcal{F}$, we can compute the similarity $\mathcal{S} \in \mathbb{R}^M$, where each element in $\mathcal{S}$ represents how likely the point belongs to the object category. Given the similarity $\mathcal{S}$, we can cluster the point clouds from the same object category into different object instances using the density-based spatial clustering of applications with noise (DBSCAN). Each clustering centroid represents one object instance.

`sel_pos`: This function selects the target instance from an input object list (obtained via `detect`) based on spatial relations, such as *far away*, *close to*, or *left*. It is necessary for instructions requiring geometric understanding. To implement this function, we invoke a VLM, which generates code to compute distances or compare coordinates to identify the target instance.

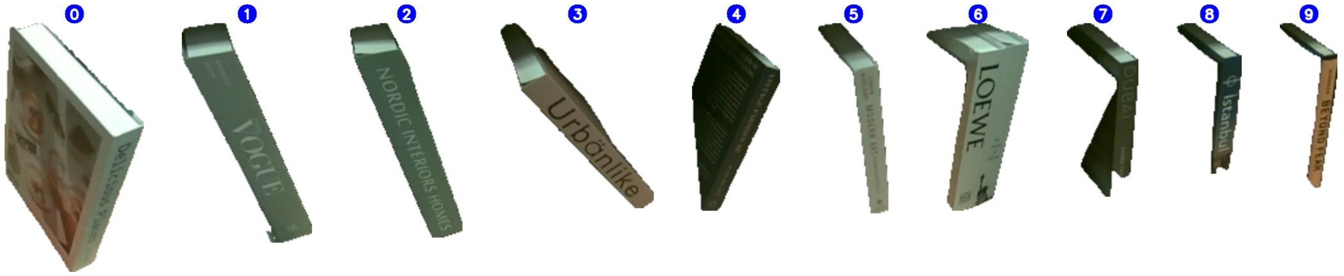


Fig. 4: **Image Input to `sel_name`.** After segmenting 3D instances and projecting them into 2D images, we concatenate the masked images and overlay instance labels. The `sel_name` function takes this composite image as input and outputs the selected instance ID. This example shows the input image for the book stowing task.

`sel_name`: Similar to `sel_pos`, but for instructions requiring semantic understanding, such as *Bob's mug* or *blue mug*. Inside this function, it passes the current observations into a VLM for instance selection. Concretely, we project 3D instances to 2D images and plot their corresponding bounding boxes. Then we feed the current observation overlayed with instance bounding boxes into VLM for instance selection.

Given the selected object instances, we project them onto 2D images and prompt Segment Anything [4] using the projected 2D bounding boxes. Next, we fuse the 2D segmentations from multiple views into 3D space to generate a 3D attention map $\mathcal{I} \in \mathbb{R}^M$, where M is the number of points. Each element in $\mathcal{I}$ indicates whether the corresponding point belongs to the task-relevant object instances. By concatenating the 3D attention map $\mathcal{I}$ with the 3D point cloud $\mathcal{P}$, we construct the state representation z for the downstream visuomotor policy. Since the generated code selects one instance from all valid detected instances (e.g., all mugs), the computed 3D attention map helps resolve the language ambiguity.

D. Visuomotor Policy Learning

Similar to Diffusion Policy [1, 82], we model our policy as Denoising Diffusion Probabilistic Models (DDPMs). Instead of predicting the action directly, we train a noise prediction network ϵ_θ conditioned on state representation z:

$$\widehat{\epsilon^k} = \epsilon_\theta(a^k, z, k), \tag{4}$$

where inputs are noisy actions a^k, current state representations $z = (\mathcal{P}, \mathcal{I})$, and denoising iterations k and outputs are the noise $\widehat{\epsilon^k}$. During training, we sample denoising step k and noise ϵ^k added to the unmodified sample a^0. Our training loss is defined as the Mean Square Error (MSE) between the added noise ϵ^k and the predicted noise:

$$\mathcal{L} = \text{MSELoss}(\epsilon^k, \widehat{\epsilon^k}). \tag{5}$$

At inference time, the policy begins with random actions a^K and denoises them over K iterations to generate the final action predictions. At each iteration, the action is updated according to the following equation:

$$a^{k-1} = \alpha\big(a^k - \gamma\epsilon_\theta(a^k, z, k) + \mathcal{N}(0, \sigma^2 I)\big), \tag{6}$$

where α, γ, and σ are hyperparameters.

In contrast to the original variant of Diffusion Policy, where z is the RGB observation, our z is a state representation, concatenating 3D point clouds $\mathcal{P}$ and 3D attention map $\mathcal{I}$. Since the 3D attention map can highlight object instances relevant to the task instruction l, we adopt a 3D attention map as an intermediate representation to bridge a high-level VLM and a low-level visuomotor policy. Therefore, our framework can leverage a VLM's reasoning capability and generate low-level actions to accomplish the task following the language instruction. We adopt PointNet++ [83] to process point cloud inputs, with additional residual connection.

To train the policy, we collect the human demonstrations using a teleoperation interface in the real world or scripted policy in the simulation. In addition, we include a lightweight annotation process to label reference DINOv2 feature for reference objects and to generate ground-truth 3D attention maps. At inference time, our system predicts actions given the current observations, instruction, and a list of reference object features in the scene.

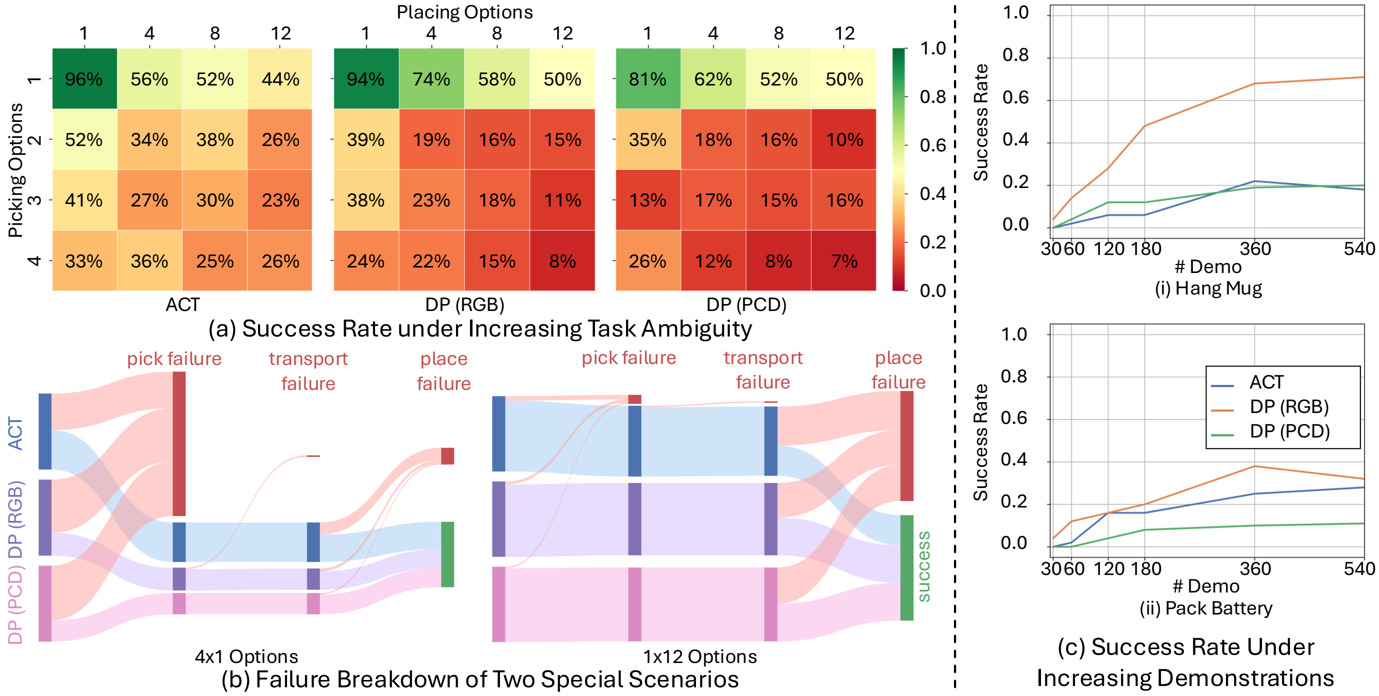
IV. Experiments

In this section, we aim to answer four questions: (1) How does the current imitation learning policy perform as task ambiguity increases, and can more data alone resolve this issue? (Section IV-B) (2) Does the 3D attention map generated by VLM-generated code align with the language instruction? (Section IV-C) (3) Is the 3D attention map a suitable representation for the downstream visuomotor policy to handle task ambiguity? (Section IV-D) (4) How well does the entire system perform in both comprehensive simulation and real-world evaluations? (Section IV-E.)

A. Experiment Setup

We use SAPIEN as the platform for large-scale simulation experiments [84]. For real-world robot experiments, we use the ALOHA system for data collection and evaluation [6], along with four RealSense cameras positioned around the workspace to capture multi-view RGB-D observations.

We consider three practical tasks: `Pack Battery`, `Hang Mug`, and `Stow Books`. These tasks involve potentially complex ambiguities, such as multiple picking and placing options in the battery packing task. In addition to collecting real-world demonstrations, we design a lightweight labeling process to generate language instructions and 3D attention maps for training the low-level policy.


(a) Success Rate under Increasing Task Ambiguity

(b) Failure Breakdown of Two Special Scenarios

(c) Success Rate Under Increasing Demonstrations

Fig. 5: **Analysis of Existing Imitation Learning Algorithms. (a)** We first evaluate three SOTA imitation learning algorithms–ACT, DP (RGB), and DP (PCD)–on the `Pack Battery` task with increasing task ambiguity. Each element in the matrix represents the method's success rate under the corresponding task setting, where columns indicate the number of empty slots and rows indicate the number of batteries to pick. *A rollout is considered successful if a battery is placed into any slot.* All policies are trained on 180 demonstrations. For simple tasks with no ambiguity (top-left entry), the high success rate confirms the validity of the baseline methods. However, as the number of picking and placing options increases, success rates decline, highlighting the vulnerability of existing methods to task ambiguity. **(b)** We further analyze failure patterns in cases with more picking options (4×1, bottom-left entry of (a)) and more placing options (1×12, top-right entry of (a)) respectively. We observe that failure primarily occurs at the task stage with the highest ambiguity, demonstrating a strong correlation between policy failure and task ambiguity. **(c)** While (a) and (b) examine existing imitation learning algorithms trained on 180 demonstrations, we also investigate whether increasing the number of demonstrations mitigates the issue. For the `Pack Battery` task, we train and test on a mixture of 1 to 4 picking options with 1 placing option. For the `Hang Mug` task, we train and test on the scene with 2 mugs for picking and 3 branches for placing. We find that adding additional demonstrations in these settings often shows diminishing returns at low success rates even with extensive demonstrations, indicating that additional training data alone may not resolve the problem.

For method evaluation, we report the task success rate, where the success criteria are determined by the information provided to the method. For methods conditioned on language or attention, we consider a rollout successful if the task is completed in the desired manner, such as successfully following the language instruction or picking the highlighted mug and placing it on the highlighted branch. For baseline imitation learning methods without additional conditioning, a rollout is considered successful if the base task is completed, regardless of the specific route or mode of execution (e.g., as long as a mug is placed on a branch). We note that this distinction results in a stricter evaluation metric for methods conditioned on additional instructional information.

B. Analysis of Existing Imitation Learning Algorithm

In this section, we aim to study how well current imitation learning algorithms can handle task ambiguities. Specifically, we consider two state-of-the-art methods, Action Chunking Transformer (ACT) [6] and Diffusion Policy (DP) [1] in comprehensive simulation evaluations. For DP, we consider two variants - DP with RGB inputs, denoted as "DP (RGB)", and DP with point cloud inputs, denoted as "DP (PCD)". We note that here we do not consider language inputs and focus on studying the low-level policy capabilities.

In our first experiment, we evaluate ACT, DP (RGB), and DP (PCD) on the `Pack Battery` task with increasing task modality, as shown in Figure 5 (a). Each entry in the matrix in represents the method's success rate under the corresponding task setting, where columns indicate the number of empty slots to place the battery and rows indicate the number of batteries to pick. A rollout is considered successful if *a* battery is placed into *any* slot. All policies are trained on 180 demonstrations. For the simplest task (i.e., top-left entry), there is only one battery to pick and one fixed placing slot, meaning a single picking option and a single placing option. For the most ambiguous task (i.e., bottom-right entry), there are up to four batteries and twelve empty slots to choose from, corresponding to four picking options and twelve placing options. Task-level ambiguity was gradually increased from the top left to the bottom right.

We observe that all baseline methods perform well in the simple task setting, demonstrating that the policy can successfully accomplish the task when no task ambiguity is present. However, as the task setting becomes more ambiguous due to an increasing number of possible choices, the performance of existing imitation learning algorithms degrades significantly. This suggests that current imitation learning algorithms struggle to handle task ambiguity.

Furthermore, we analyze the failure patterns of two specific cases–four picking options with one placing option (i.e., 4×1 options) and one picking option with twelve placing options (i.e., 1×12 options)–as shown in Figure 5 (b). As the number of picking options increases, ACT and DP struggle to execute the picking skill. Similarly, as the number of placement options increases, most failures occur during the placement stage of the task. The observed correlation between (i) increased task ambiguity and (ii) declining task success rates further underscores the limitations of existing imitation learning algorithms in handling task ambiguity.

In addition, we investigate whether increasing the number of demonstrations can help existing imitation learning algorithms handle task ambiguity. We conduct experiments on the `Pack Battery` and `Hang Mug` tasks in simulation. For the `Pack Battery` task, we train and test on a mixture of 1 to 4 picking options with 1 placing option. For the `Hang Mug` task, we train and test on a scene with 2 mugs for picking and 3 branches for placing. The number of demonstrations is increased from 30 to 540 episodes. While the performance of ACT and DP initially improves, they generally show diminishing returns while success rate is still low, and in some cases plateaus as the number of demonstrations further increases, suggesting that additional demonstrations may not effectively resolve task ambiguity.

C. Evaluation of 3D Attention Maps

In this section, we evaluate the pipeline from language instructions to 3D attention maps across different scenes and instructions. Figure 6 illustrates our system's performance in various scenarios. We observe that for a simple instruction such as "Put the rightmost battery in the slot on the left column," the 3D attention maps correctly highlight the intended battery instance and slot position. The VLM-generated code can also perform zero-shot interpretation of language exhibiting more complicated logical structures, such as self-repairing phrases such as "Hang the red mug on the top branch. Sorry, the green mug." The generated 3D attention maps correctly highlight the green mug and top branch. Furthermore, even with ambiguous commands like "Hang a mug on a branch" (without specifying a particular mug or branch), our system autonomously selects and highlights appropriate objects. These results demonstrate the system's ability to handle ambiguous or vague instructions while highlighting its semantic understanding and capability to generate accurate 3D attention maps across diverse instructions and scenarios.

In addition, we build a benchmark in the simulation to quantitatively evaluate the language-to-3D attention pipeline, which can automatically generate scenes, prompts, and corresponding ground truth 3D attention maps. We measure the distance between ground truth 3D attention maps and generated 3D attention maps, and a test is considered successful if they are close enough.

Scene	Hang Mug	Pack Battery	Total
Ours	97/100	94/100	191/200

Table I: Attention Quantitative Evaluation. We quantitatively evaluate the pipeline from language instructions to 3D attention maps in simulation. Our results demonstrate that our pipeline effectively attends to task-relevant areas.

Table I summarizes our quantitative evaluation of the pipeline from language to attention. Our method effectively leverages the powerful visual-semantic understanding capabilities of VLMs and benefits from explicit spatial relation reasoning using 3D representations. Additional analysis and visualizations of 3D attention failure cases are provided in the supplementary materials.

D. Evaluation of Attention-Conditioned Diffusion Policy

In this section, we investigate whether 3D attention is a suitable representation for visuomotor policy learning and evaluate the pipeline from 3D attention maps to low-level actions. We first evaluate our method by varying the number of demonstrations on the `Pack Battery` task in simulation, as shown in Figure 7 (a). We train and test on a scene consisting of a mixture of 1 to 4 picking options and 1 placing option. A policy rollout is considered successful only if it completes the task as specified by the given groundtruth 3D attention map. Figure 7 (a) shows that our system's success rate reaches (>90%) at approximately 120 demos, indicating that our method can effectively handle task ambiguity. Additionally, we increase task ambiguity and observe its effect on the success rate, as shown in Figure 7 (b). Each entry in the matrix represents the success rate under the corresponding task setting, where rows indicate the number of picking options and columns indicate the number of placing options. Figure 7 (b) demonstrates that our system is not significantly affected by task ambiguity and performs well across both simple and complex task settings. These results confirm that the 3D attention map is a robust representation for downstream visuomotor policy learning in ambiguous task scenarios.

Additionally, we stress-test our visuomotor policy on the `Pack Battery` task across unseen scenarios in simulation. In Figure 7 (c), each entry represents the success rate of the trained policy in different testing environments, where rows indicate training environments and columns indicate testing environments. For example, the second row (i.e., 1×2) corresponds to the training scenario with one picking option and two placing options, while the third column (i.e., 1×3) represents the testing scenario with one picking option and three placing options. First, we observe that the success rate along the diagonal is high, validating the expected performance pattern where policies perform well on their original training scenarios. Second, while training on a 1x1 scenario does not generalize to scenarios with multiple placing options, the generalization of CodeDiffuser quickly improves after seeing more than one placing option at training time. The 3D

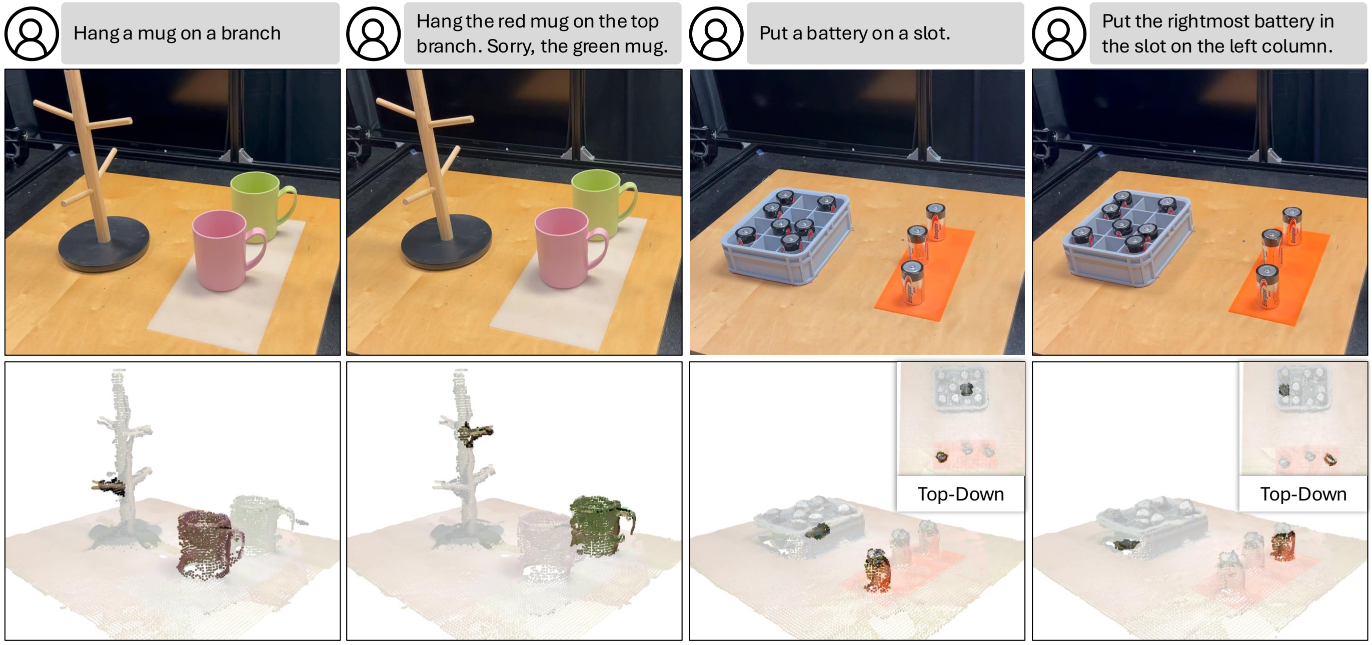

Hang a mug on a branch
Hang the red mug on the top branch. Sorry, the green mug.
Put a battery on a slot.
Put the rightmost battery in the slot on the left column.
Top-Down
Top-Down

Fig. 6: **3D Attention Maps Visualization.** We visualize the 3D attention maps for corresponding instructions and scenarios. First, our 3D attention maps successfully highlight the correct object instances even under ambiguous instructions, such as "Hang a mug on a branch" or "Put a battery on a slot," without specifying the instance. Furthermore, as instructions become more complex and/or specific, the 3D attention maps continue to attend to the correct instances, accurately matching the given instructions.

attention module enables the policy to focus on task-relevant visual regions, thereby enhancing its ability to generalize across diverse and previously unseen task settings.

E. Evaluation of the Entire System

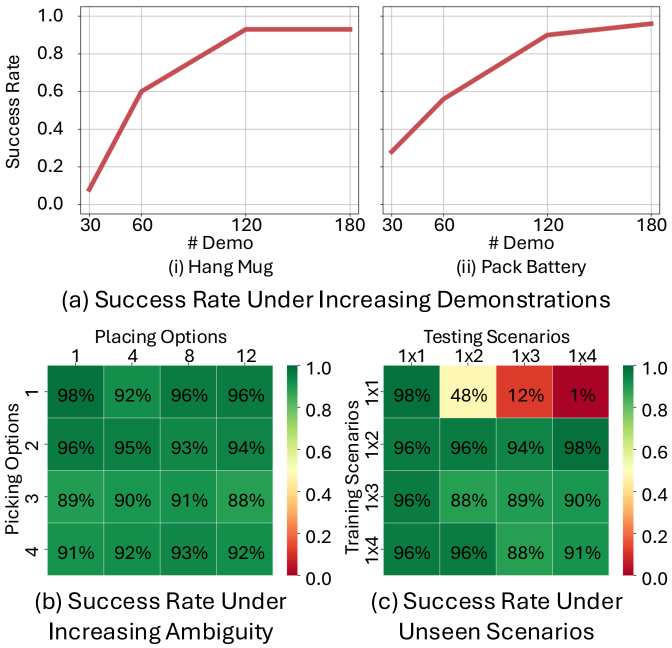
In this section, we evaluate the entire robotic manipulation framework, from language to low-level actions, through both quantitative and qualitative analyses. In the simulation, we collect 180 training demonstrations for each task. We consider two simulation tasks: `Hang Mug` and `Pack Battery`, which involve language ambiguity, contact-rich manipulation, and multi-object interactions. The `Hang Mug` task requires picking between two mugs and placing them on one of four branches, while the `Pack Battery` task involves picking from four batteries and placing them into one of twelve slots. For the simulation experiments, we compare our method against the following baselines:

- **Ours with 2D Attention**: Instead of mapping the multi-view RGBD observation into 3D space, this baseline uses a 2D attention mechanism to segment objects of interest. The masked observation is then input into visuomotor policy.
- **Ours without Residual Connection**: In our policy, we include a residual connection in PointNet++ for visual feature extraction. This baseline ablates the residual connection to evaluate its contribution to performance.
- **Lang-DP (RGB)**: This baseline extends DP (RGB) by conditioning the policy on language using a frozen CLIP encoder. The extracted language features are concatenated with visual features to condition the diffusion policy.
- **Lang-DP (PCD)**: Similar to Lang-DP (RGB), this baseline adds language conditioning to DP (PCD) using a CLIP-based language encoder.
- **Lang-ACT**: This baseline augments the original ACT framework with language features, similar to Lang-DP.
- **Lang-ACT with 3D Attention**: Unlike the original ACT, which uses multi-view RGB observations, this baseline inputs 3D attention maps into Lang-ACT to assess whether the attention module consistently improves the performance of base imitation learning algorithms.

The prompts are generated from randomly selected descriptive components, such as "right," "furthest," and "blue." These components are incorporated into templates such as "Put the blue mug into the furthest slot." More specifically, all prompts can be categorized into four types:

- **Prompt without slackness**: All objects are strictly specified, such as "Hang the left-most mug on the top branch."
- **Prompt with full slackness**: No objects are strictly specified, such as "Hang a mug on a branch."
- **Prompt with picked-object slackness**: The placement locations (e.g., branch or slot) are strictly specified, while the objects being placed are not, such as "Hang a mug on the left-most branch."
- **Prompt with placed-object slackness**: The objects being placed (e.g., mug or battery) are strictly specified, while the placement locations are not, such as "Hang the blue mug on a branch."

Figure 8 (b) summarizes the quantitative results from the simulation experiments. From this table, we draw the following conclusions: (1) Adding the attention module significantly enhances the policy's ability to accomplish tasks involving linguistic ambiguity. Compared to Lang-DP (PCD), our method leverages VLM to interpret language instructions and compute



(a) Success Rate Under Increasing Demonstrations

(b) Success Rate Under Increasing Ambiguity

(c) Success Rate Under Unseen Scenarios

Fig. 7: **Analysis of Attention-Conditioned Policy.** (a) We first examine the performance of the attention-conditioned policy under varying numbers of demonstrations. *A rollout is considered as successful if the policy accomplishes the task as specified by the 3D attention maps*, a stricter criteria than that of Fig. 5. The training and testing scenarios consist of a mixture of 1 to 4 picking options with 1 placing option. The success rate curve indicates that, given a sufficient number of demonstrations, our attention-conditioned policy converges to a high success rate. (b) We then analyze the performance of the attention-conditioned policy under increasing task ambiguity. Similar to Figure 5, each element represents the success rate, with columns indicating the number of placing options and rows indicating the number of picking options. The success rate remains consistently high as task ambiguity increases, demonstrating that 3D attention serves as an effective representation for the downstream visuomotor policy. (c) Additionally, we find 3D attention improves policy generalization. When trained on scenarios with lower ambiguity, such as 1 picking option with 2 placing options (i.e., 1×2 in the row), the policy generalizes well to scenarios with greater ambiguity, such as those involving 3 or 4 placing options (i.e., 1×3 and 1×4 in the column).

3D attention maps, resulting in a substantial performance improvement from 5.5% to 86.5%. (2) The 3D attention maps can also be integrated into other base imitation learning algorithms to improve performance. For instance, comparing Lang-ACT with 3D Attention to Lang-ACT, we observe a significant performance increase from 6% to 61%. (3) The attention mechanism is also effective for 2D representations. In contrast to Lang-DP (RGB), our method, which incorporates a similar pipeline from language instructions to 2D attention, achieves a performance improvement from 12% to 84.5%. (4) Our ablation study demonstrates that incorporating the residual connection into PointNet++ improves performance from 61% to 86.5%. This enhancement is attributed to the residual connection's ability to better propagate attention information into the visuomotor policy, thereby improving trajectory prediction.

In addition, we evaluate the entire system on the `Hang Mug`, `Pack Battery`, and `Stow Book` tasks in the real world. We collect 150 demonstrations for each real-world task. For the `Stow Book` task–a challenging task that is difficult to simulate due to its contact-rich nature–we test on scenes with two available slots for placement. We note that while using 2D attention achieves similar performance to using 3D attention maps, we adopt 3D attention maps for real-world tasks due to their robustness to environmental factors, as observed in DP3 and GenDP [34, 40]. We compare our method to Lang-DP (RGB) and Lang-DP (Colored PCD), where the point cloud is colored based on RGB observations.

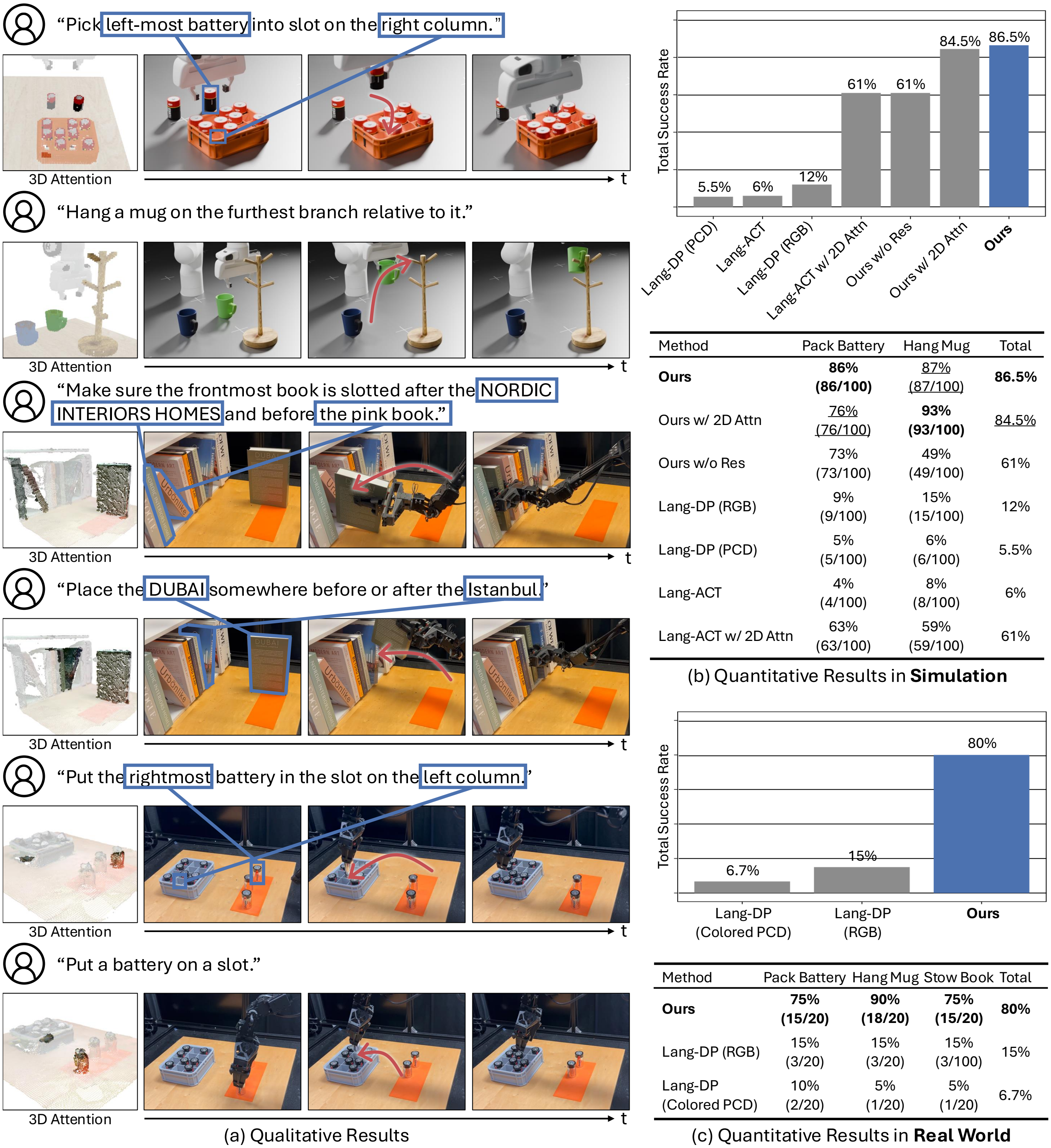
Figure 8 (c) summarizes the quantitative evaluation results from the real-world experiments. We find that our policy consistently outperforms the baselines by leveraging VLM-generated code as an interpretable and executable intermediate representation, effectively utilizing the visual-semantic reasoning capabilities of the VLM.

Figure 8 (a) presents our qualitative evaluation. Our policy effectively handles task instructions with varying degrees of specificity. For instance, the instruction "Put a battery on a slot." is ambiguous regarding the target battery and slot. Our system interprets this ambiguous instruction, selects a specific instance, and successfully executes the task. In contrast, another instruction with a similar initial object configuration explicitly specifies the target battery and slot. Thanks to the VLM's visual-semantic reasoning capabilities, our system correctly interprets this precise instruction, identifies the appropriate task-relevant locations, and successfully completes the task.

In addition, we analyze the common failure cases of our method, as shown in Figure 9. We break down the failure pattern according to the pipeline modules, including code generation failure, perception failure, and task execution failure. We observe that the majority of failure focuses on task execution, while the code generation and perception are relatively stable.

V. Conclusion

Language ambiguity is a common challenge in robotic manipulation, such as determining which mug to pick and where to place it for the instruction "Hang a mug on a branch." Existing imitation learning algorithms typically employ end-to-end models that jointly interpret high-level semantic information and generate low-level actions, often resulting in suboptimal performance. In this work, we address this challenge by introducing a novel robotic manipulation framework that utilizes VLM-generated code as an executable and interpretable intermediate representation. The generated code interfaces with perception APIs to compute 3D attention maps using VFMs, which are then used for downstream visuomotor policy execution. Our modular design leverages both the visual-semantic understanding capabilities of VLMs and the smooth trajectory prediction of low-level policies. In our experiments, we first identify the key limitations of existing imitation learning algorithms. We then conduct a comprehensive evaluation of our method in both simulation and real world, and study the pipeline from language to 3D attention maps, the pipeline from 3D attention maps to low-level actions, and the entire system respectively. We demonstrate CodeDiffuser's effectiveness in challenging robotic tasks


"Pick left-most battery into slot on the right column."
3D Attention
t
"Hang a mug on the furthest branch relative to it."
3D Attention
t
"Make sure the frontmost book is slotted after the NORDIC INTERIORS HOMES and before the pink book."
3D Attention
t
"Place the DUBAI somewhere before or after the Istanbul."
3D Attention
t
"Put the rightmost battery in the slot on the left column."
3D Attention
t
"Put a battery on a slot."
3D Attention
t
Total Success Rate
5.5%
6%
12%
61%
61%
84.5%
86.5%
Lang-DP (PCD)
Lang-ACT
Lang-DP (RGB)
Lang-ACT w/ 2D Attn
Ours w/o Res
Ours w/ 2D Attn
Ours
Total Success Rate
6.7%
15%
80%
Lang-DP (Colored PCD)
Lang-DP (RGB)
Ours

(a) Qualitative Results

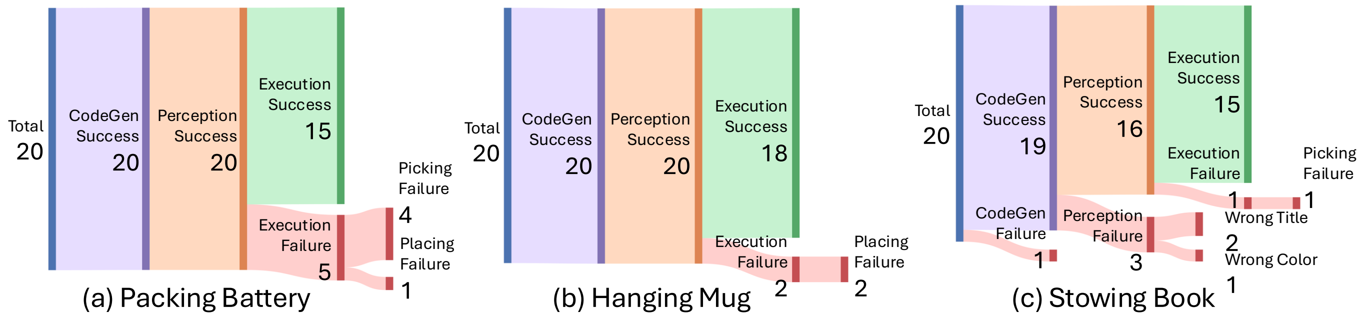
Method	Pack Battery	Hang Mug	Total
Ours	**86% (86/100)**	87% (87/100)	**86.5%**
Ours w/ 2D Attn	76% (76/100)	**93% (93/100)**	84.5%
Ours w/o Res	73% (73/100)	49% (49/100)	61%
Lang-DP (RGB)	9% (9/100)	15% (15/100)	12%
Lang-DP (PCD)	5% (5/100)	6% (6/100)	5.5%
Lang-ACT	4% (4/100)	8% (8/100)	6%
Lang-ACT w/ 2D Attn	63% (63/100)	59% (59/100)	61%

(b) Quantitative Results in **Simulation**

Method	Pack Battery	Hang Mug	Stow Book	Total
Ours	**75% (15/20)**	**90% (18/20)**	**75% (15/20)**	**80%**
Lang-DP (RGB)	15% (3/20)	15% (3/20)	15% (3/100)	15%
Lang-DP (Colored PCD)	10% (2/20)	5% (1/20)	5% (1/20)	6.7%

(c) Quantitative Results in **Real World**

Fig. 8: **Evaluation of Entire System.** (a) We qualitatively evaluate the entire pipeline, from language instructions to low-level actions, demonstrating how our system interprets semantic meanings from abstract instructions. Given similar initial configurations, our system can react appropriately to different instructions, showcasing its semantic understanding capability. (b) In the simulation, we compare our method against various baselines on both `Pack Battery` and `Hang Mug` tasks. A policy rollout is considered successful if it manipulates the object as specified by the instruction. Our results show that our system significantly outperforms baseline methods, highlighting its ability to comprehend high-level semantic information and execute complex tasks effectively. (c) We further validate our approach in real-world experiments using the same evaluation metric. Our system successfully performs challenging tasks involving contact-rich manipulation and multi-object interactions, consistently surpassing baseline methods in performance.



(a) Packing Battery (b) Hanging Mug (c) Stowing Book

Fig. 9: **System Failure Breakdown.** We categorize the failure patterns in the real-world experiments into code generation failures, perception failures, and execution failures. Our results indicate that the majority of failures occur during task execution, while code generation and perception remain relatively stable.

involving task-level ambiguity, contact-rich 6-DoF manipulation, and multi-object interactions.

Limitations: The proposed method relies on VLMs and VFMs to generate codes, detect objects, and compute 3D attention maps. Therefore, the performance of our system is bounded by VLMs and VFMs. In the future, having more advanced VLMs and VFMs can benefit our system and make our system more robust. In addition, our perception APIs currently operate on an object level, which can pose challenges in generalizing to instructions involving deformable objects, such as "grab the top of the circle of dough". In addition, the additional manual annotation process is practically challenging for large-scale training. Scaling this method to larger datasets by simplifying the manual annotation process is a promising direction for future work.

ACKNOWLEDGEMENT

This work is partially supported by the Toyota Research Institute (TRI), the Sony Group Corporation, Google, and Dalus AI. We greatly appreciate the NCSA for providing computing resources. This article solely reflects the opinions and conclusions of its authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

REFERENCES

[1] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[2] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.

[3] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[5] Tony Z Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Seyed Kamyar Seyed Ghasemipour, Chelsea Finn, and Ayzaan Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *8th Annual Conference on Robot Learning*, 2024.

[6] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[7] Zipeng Fu, Tony Z Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

[8] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[9] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. *arXiv preprint arXiv:2407.10353*, 2024.

[10] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2018.

[11] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. *arXiv preprint arXiv:2306.14896*, 2023.

[12] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[13] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport:

What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[14] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *6th Annual Conference on Robot Learning (CoRL)*, 2022.

[15] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *arXiv preprint arXiv:2108.03298*, 2021.

[16] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2):492–499, 2019.

[17] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[18] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017.

[19] Lucy Xiaoyang Shi, Archit Sharma, Tony Z Zhao, and Chelsea Finn. Waypoint-based imitation learning for robotic manipulation. *arXiv preprint arXiv:2307.14326*, 2023.

[20] Norman Di Palo and Edward Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024.

[21] Norman Di Palo and Edward Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. *arXiv preprint arXiv:2402.13181*, 2024.

[22] Bowen Wen, Wenzhao Lian, Kostas Bekris, and Stefan Schaal. You only demonstrate once: Category-level manipulation from single visual demonstration. *arXiv preprint arXiv:2201.12716*, 2022.

[23] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[24] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[25] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[26] Aditya Ganapathi, Pete Florence, Jake Varley, Kaylee Burns, Ken Goldberg, and Andy Zeng. Implicit kinematic policies: Unifying joint and cartesian action spaces in end-to-end robot learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2656–2662. IEEE, 2022.

[27] Pete Florence, Corey Lynch, Andy Zeng, Oscar Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. *Conference on Robot Learning (CoRL)*, 2021.

[28] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. *Advances in neural information processing systems*, 35:22955–22968, 2022.

[29] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

[30] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on Machine Learning*, 2011. URL https://api.semanticscholar.org/CorpusID:2178983.

[31] Igor Mordatch. Concept learning with energy-based models. *arXiv preprint arXiv:1811.02486*, 2018.

[32] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. https://octo-models.github.io, 2023.

[33] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[34] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

[35] Justin Kerr, Chung Min Kim, Mingxuan Wu, Brent Yi, Qianqian Wang, Ken Goldberg, and Angjoo Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction, 2024. URL https://arxiv.org/abs/2409.18121.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[37] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino

with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[38] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16901–16911, 2024.

[39] Yixuan Wang, Mingtong Zhang, Zhuoran Li, Tarik Kelestemur, Katherine Rose Driggs-Campbell, Jiajun Wu, Li Fei-Fei, and Yunzhu Li. D^{3}fields: Dynamic 3d descriptor fields for zero-shot generalizable rearrangement. In *8th Annual Conference on Robot Learning*, 2024.

[40] Yixuan Wang, Guang Yin, Binghao Huang, Tarik Kelestemur, Jiuguang Wang, and Yunzhu Li. Gendp: 3d semantic fields for category-level generalizable diffusion policy. In *8th Annual Conference on Robot Learning*, 2024.

[41] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *8th Annual Conference on Robot Learning*, 2024.

[42] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning*, pages 540–562. PMLR, 2023.

[43] Ri-Zhao Qiu, Yafei Hu, Ge Yang, Yuchen Song, Yang Fu, Jianglong Ye, Jiteng Mu, Ruihan Yang, Nikolay Atanasov, Sebastian Scherer, and Xiaolong Wang. Learning generalizable feature fields for mobile manipulation. *arXiv preprint arXiv:2403.07563*, 2024.

[44] Yanjie Ze, Ge Yan, Yueh-Hua Wu, Annabella Macaluso, Yuying Ge, Jianglong Ye, Nicklas Hansen, Li Erran Li, and Xiaolong Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on Robot Learning*, pages 284–301. PMLR, 2023.

[45] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[46] Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *7th Annual Conference on Robot Learning*, 2023.

[47] Xingyu Lin, John So, Sashwat Mahalingam, Fangchen Liu, and Pieter Abbeel. Spawnnet: Learning generalizable visuomotor skills from pre-trained network. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4781–4787. IEEE, 2024.

[48] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028. IEEE, 2024.

[49] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023.

[50] Hanxiao Jiang, Binghao Huang, Ruihai Wu, Zhuoran Li, Shubham Garg, Hooshang Nayyeri, Shenlong Wang, and Yunzhu Li. Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. *arXiv preprint arXiv:2402.15487*, 2024.

[51] Nathan Hughes, Yun Chang, and Luca Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization. *arXiv preprint arXiv:2201.13360*, 2022.

[52] Dominic Maggio, Yun Chang, Nathan Hughes, Matthew Trang, Dan Griffith, Carlyn Dougherty, Eric Cristofalo, Lukas Schmid, and Luca Carlone. Clio: Real-time task-driven open-set 3d scene graphs. *arXiv preprint arXiv:2404.13696*, 2024.

[53] Nathan Hughes, Yun Chang, Siyi Hu, Rajat Talak, Rumaia Abdulhai, Jared Strader, and Luca Carlone. Foundations of spatial perception for robotics: Hierarchical representations and real-time systems. *The International Journal of Robotics Research*, page 02783649241229725, 2024.

[54] Jared Strader, Nathan Hughes, William Chen, Alberto Speranzon, and Luca Carlone. Indoor and outdoor 3d scene graph generation via language-enabled spatial ontologies. *IEEE Robotics and Automation Letters*, 2024.

[55] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023.

[56] Huy Ha and Shuran Song. Semantic abstraction: Open-world 3d scene understanding from 2d vision-language models. *arXiv preprint arXiv:2207.11514*, 2022.

[57] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024.

[58] Yinpei Dai, Run Peng, Sikai Li, and Joyce Chai. Think, act, and ask: Open-world interactive personalized robot navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3296–3303. IEEE, 2024.

[59] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[60] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn,

Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[61] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[62] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[63] Haoxu Huang, Fanqi Lin, Yingdong Hu, Shengjie Wang, and Yang Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. *arXiv preprint arXiv:2403.08248*, 2024.

[64] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[65] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.

[66] Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, et al. Video language planning. *arXiv preprint arXiv:2310.10625*, 2023.

[67] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36: 20482–20494, 2023.

[68] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024.

[69] Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically grounded vision-language models for robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12462–12469. IEEE, 2024.

[70] Yanwei Wang, Tsun-Hsuan Wang, Jiayuan Mao, Michael Hagenow, and Julie Shah. Grounding language plans in demonstrations through counterfactual perturbations. *arXiv preprint arXiv:2403.17124*, 2024.

[71] Joy Hsu, Jiayuan Mao, and Jiajun Wu. Ns3d: Neuro-symbolic grounding of 3d objects and relations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2614–2623, 2023.

[72] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[73] Jiafei Duan, Wentao Yuan, Wilbert Pumacay, Yi Ru Wang, Kiana Ehsani, Dieter Fox, and Ranjay Krishna. Manipulate-anything: Automating real-world robots using vision-language models. *arXiv preprint arXiv:2406.18915*, 2024.

[74] Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. A2nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *arXiv preprint arXiv:2308.07997*, 2023.

[75] Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7694–7701. IEEE, 2024.

[76] Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.

[77] Yafei Hu, Quanting Xie, Vidhi Jain, Jonathan Francis, Jay Patrikar, Nikhil Keetha, Seungchan Kim, Yaqi Xie, Tianyi Zhang, Zhibo Zhao, et al. Toward general-purpose robots via foundation models: A survey and meta-analysis. *arXiv preprint arXiv:2312.08782*, 2023.

[78] Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *arXiv preprint arXiv:2312.07843*, 2023.

[79] Kento Kawaharazuka, Tatsuya Matsushima, Andrew Gambardella, Jiaxian Guo, Chris Paxton, and Andy Zeng. Real-world robot applications of foundation models: A review. *arXiv preprint arXiv:2402.05741*, 2024.

[80] Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*, 2023.

[81] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[82] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[83] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural*

information processing systems, 30, 2017.

[84] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.