VoxAct-B: Voxel-Based Acting and Stabilizing Policy for Bimanual Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Bimanual manipulation is critical to many robotics applications. In contrast to single-arm manipulation, bimanual manipulation tasks are challenging due to higher-dimensional action spaces. Prior works leverage large amounts of data and primitive actions to address this problem, but may suffer from sample inefficiency and limited generalization across various tasks. To this end, we propose VoxAct-B, a language-conditioned, voxel-based method that leverages Vision Language Models (VLMs) to prioritize key regions within the scene and reconstruct a voxel grid. We provide this voxel grid to our bimanual manipulation policy to learn acting and stabilizing actions. This approach enables more efficient policy learning from voxels and is generalizable to different tasks. In simulation, we show that VoxAct-B outperforms strong baselines on fine-grained bimanual manipulation tasks. Furthermore, we demonstrate VoxAct-B on real-world `Open Drawer` and `Open Jar` tasks using two UR5s. Code, data, and videos will be available at https://voxact-b.github.io.

1 Introduction

Bimanual manipulation is essential for robotics tasks, such as when objects are too large to be controlled by one gripper or when one arm stabilizes an object of interest to make it simpler for the other arm to manipulate [1]. In this work, we focus on asymmetric bimanual manipulation. Here, "asymmetry" refers to the functions of the two arms, where one is a *stabilizing* arm, while the other is the *acting* arm. Asymmetric tasks are common in household and industrial settings, such as cutting food, opening bottles, and packaging boxes. They typically require two-hand coordination and high-precision, fine-grained manipulation, which are challenging for current robotic manipulation systems. To tackle bimanual manipulation, some methods [2, 3] train policies on large datasets, and some exploit primitive actions [4, 5, 6, 7, 8, 9, 10]. However, they are generally sample inefficient, and using primitives can hinder generalization to different tasks.

To this end, we propose VoxAct-B, a novel voxel-based, language-conditioned method for bimanual manipulation. Voxel representations, when coupled with discretized action spaces, can increase sample efficiency and generalization by introducing spatial equivariance into a learned system, where transformations of the input lead to corresponding transformations of the output [11]. However, processing voxels is computationally demanding [12, 13]. To address this, we propose utilizing VLMs to focus on the most pertinent regions within the scene by cropping out less relevant regions. This substantially reduces the overall physical dimensions of the areas used to construct a voxel grid, enabling an increase in voxel resolution without incurring computational costs. To our knowledge, this is the first study to apply voxel representations in bimanual manipulation.

We also employ language instructions and VLMs to determine the roles of each arm: whether they are *acting* or *stabilizing*. For instance, in a drawer-opening task, the orientation of the drawer and the position of the handle affect which arm is more suitable for opening the drawer (acting) and which is better for holding it steady (stabilizing). We use VLMs to compute the pose of the object of interest relative to the front camera and to decide the roles of each arm. Then, we provide appropriate language instructions to the bimanual manipulation policy to control the acting and stabilizing arms.

Submitted to the 8th Conference on Robot Learning (CoRL 2024). Do not distribute.

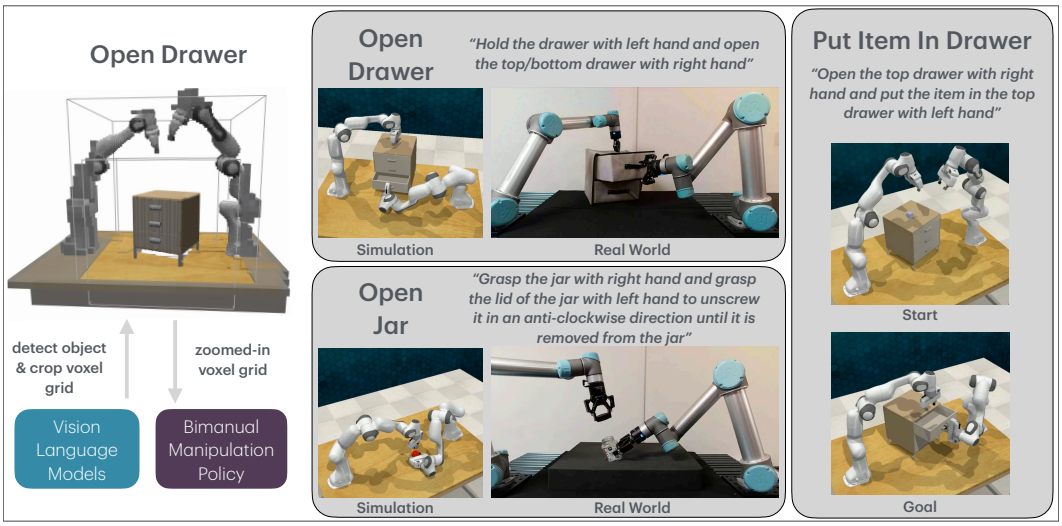

Figure 1: VoxAct-B uses voxel representations and language to perform bimanual manipulation with 6-DoF manipulation from both arms. We test three language-conditioned bimanual tasks in simulation and two (`Open Drawer` and `Open Jar`) on a real-world setup with two UR5s. The prompt for `Open Drawer` assumes the left arm is stabilizing and the right arm is acting, while the reverse is true for the `Open Jar` prompt.

We extend the RLBench [14] benchmark to support bimanual manipulation. We introduce an asymmetric bimanual version of `Open Drawer`, `Open Jar`, and `Put Item in Drawer` tasks. VoxAct-B outperforms strong baselines, such as ACT [3], Diffusion Policy [15], VoxPoser [16], and PerAct [11], by a large margin. We also validate our approach on a real-world bimanual manipulation setup with two UR5s on `Open Drawer` and `Open Jar`. See Figure 1 for an overview.

The contributions of this paper include:

- VoxAct-B, a novel method for bimanual manipulation which uses VLMs to reduce the size of a voxel grid for learning with a modified, downstream voxel-based behavior cloning method [11].
- A suite of vision-language bimanual manipulation tasks, extended from RLBench [14].
- Simulation experiments indicating that VoxAct-B achieves state-of-the-art results on these tasks.
- Demonstrations of VoxAct-B on a real-world bimanual manipulation setup with two UR5s.

2 Related Work

Bimanual Manipulation. There has been much prior work in bimanual manipulation for folding cloth [5, 8, 17, 18, 19, 20, 21], cable untangling [6], scooping [9], bagging [22, 23, 24], throwing [25], catching [26], and untwisting lids [27]. Other works study bimanual manipulation with dexterous manipulators [28, 29, 30, 31] or mobile robots [32]. In contrast to these works, our focus is on a *general approach* to bimanual manipulation with parallel-jaw grippers on fixed-base manipulators. Works that study general approaches for bimanual manipulation include [2, 4, 7, 33], which use primitive actions or skills to reduce the search space across actions. Other general approaches focus on orthogonal tools such as interaction primitives [34] or screw motions [35]. Recently, Zhou et al. [3] introduced another general approach, based on "action chunking" to learn high-frequency controls with closed-loop feedback and applied their method on multiple asymmetric bimanual manipulation tasks using low-cost hardware. Other works extended this by either incorporating novel imitation learning algorithms [36] or enhancing the hardware itself [37, 38]. However, these works may still require substantial training data and lack spatial equivariance for generalization. In closely-related work, Grannen et al. [39] decouple a system into stabilizing and acting arms to enable sample-efficient bimanual manipulation with simplified data collection. While effective, this formulation predicts top-down keypoints and was not tested with 6-DoF manipulation. In contrast, our method supports 6-DoF manipulation for bimanual manipulation tasks.

Action Space Representation. For 2D manipulation, prior works have shown the benefits of action representations based on spatial action maps [21, 40, 41, 42, 43, 44, 45], including in bimanual contexts [10, 21], where neural networks directly predict 2D "images" that indicate desired locations for the action. Compared to directly regressing the action location, using spatial action maps better handles multimodality and has 2D equivariance, where translations and rotations of the input image map to similar transformations of the output action. Recent works have extended this idea to support 3D spatial action maps, which classify an action's location as a 3D point in the robot's workspace, and thus maintain spatial equivariance. For example, PerAct [11] is a language-conditioned behavioral cloning agent that takes voxel grids as input and outputs 6-DoF actions. While PerAct achieved state-of-the-art performance on RLBench, it has a high computational cost due to processing voxels. Follow-up works, such as RVT [12] and Act3D [13], have reduced the computational cost of PerAct by avoiding voxel representations but often need multiple views of the scene to achieve optimal performance and may be less interpretable compared to a voxel grid that contains a 3D spatial action map. These prior works have also not been applied to bimanual manipulation. In this work, we retain the spatial equivariance benefits of voxel representations but reduce the cost of processing voxels by "zooming" into part of the voxel grid. This is similar to the intent of C2F-ARM [46], but we use the knowledge in VLMs to determine the most relevant regions in the voxel grid.

LLMs and VLMs for Robotics. LLMs and VLMs, such as GPT-4 [47], Llama 2 [48], and Gemini [49], have revolutionized natural language processing, computer vision, and robotics due to their strong reasoning and semantic understanding capabilities. Consequently, recent work has integrated them in robotics and embodied AI agents, typically as a high-level planner [50, 51, 52, 53], which may also produce code for a robot to execute [54, 55, 56]. We defer the reader to [57, 58, 59] for representative surveys. Among the most relevant prior works, Huang et al. [16] propose VoxPoser, which uses pre-trained LLMs and VLMs to compose 3D affordance maps and 3D constraint maps, which are then used with motion planning to generate trajectories for robotic manipulation. By leveraging LLMs and VLMs, VoxPoser can generalize to open-set instructions and objects. However, as we later demonstrate in experiments, VoxPoser can struggle with tasks that require high precision and contact. In this work, we demonstrate how to use VLMs to effectively process the input of PerAct for bimanual manipulation, obtaining the generalization benefits of VLMs with the precision capabilities of PerAct. In recent and near-concurrent work, Varley et al. [60] also uses VLMs for bimanual manipulation. Our work differs in that we do not fix the roles of each arm; we use an off-the-shelf VLM [61] without any fine-tuning, and we do not use a skills library.

3 Problem Statement

Given access to a pre-trained VLM and expert demonstrations, the objective is to produce a bimanual policy π for a variety of language-conditioned manipulation tasks. We assume a flat workspace with two fixed-base robot manipulators, each with a parallel-jaw gripper. A policy π controls both arms by producing actions $a_t = (a_t^s, a_t^a)$ at each time step t, where a_t^s and a_t^a follow [39] and refer to the *stabilizing* and *acting* arm actions, respectively. For simplicity, we suppress the time t when the distinction is unnecessary. We use the low-level action representation from PerAct [11] with $a^s = (a^s_{\text{pose}}, a^s_{\text{open}}, a^s_{\text{collide}})$ and $a^a = (a^a_{\text{pose}}, a^a_{\text{open}}, a^a_{\text{collide}})$. These specify each arm's 6-DOF gripper pose, its gripper open state, and whether a motion planner for the arms used collision avoidance to reach an intermediate pose. We assume task-specific demonstrations $\mathcal{D}_\ell = \{\zeta_1, \zeta_2, \ldots, \zeta_n\}$ and two common language commands ℓ_{as} and ℓ_{sa}, where as denotes the left arm as acting and right arm as stabilizing, and vice versa for sa. Each demonstration consists of a set of keyframes extracted from a sequence of continuous actions paired with observations. We adapt the keyframe extraction function from [11] by including keyframes that have an action with near-zero joint velocities and unchanged gripper open state for acting and stabilizing arms. The observation at each time is the 3D voxel grid $\mathbf{v}$ of dimension $(L \times W \times H)$, where we use $\mathbf{v}[x, y, z]$ to denote an individual voxel at coordinates (x, y, z). The robot also receives the language command $\mathbf{l} \in \{\ell_{as}, \ell_{sa}\}$, which is fixed for all time steps in an *episode*, where the robot interacts with the environment for up to T time steps. An episode terminates with a task-dependent success criteria or failure (if otherwise).

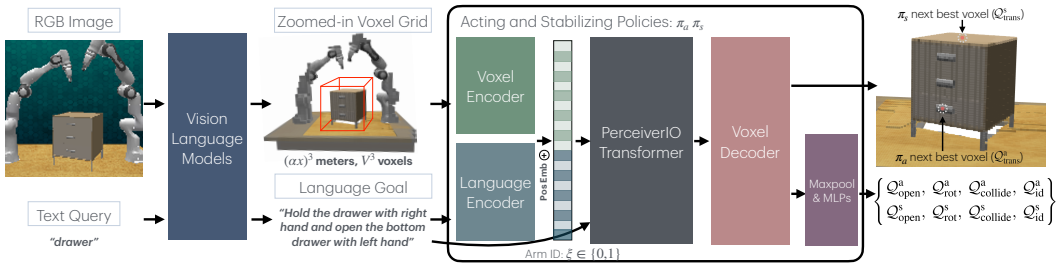

Figure 2: Overview of VoxAct-B. Given RGB-D images and a language goal, we input an RGB image from the front camera and a text query extracted from the language goal into the Vision Language Models (VLMs). The VLMs output the pose of the object of interest with respect to the front camera. This information determines the language goal and the roles of each arm (i.e., *acting* or *stabilizing*). Additionally, we use the object's position with the RGB-D images to reconstruct a voxel grid that spans αx^3 meters of the workspace using V^3 voxels. The zoomed-in voxel grid, the language goal, proprioception data of both robot arms, and an arm ID are provided to an acting policy π_a and a stabilizing policy π_s. The policies predict the discretized pose of the next best voxel, gripper open action, collision avoidance flag, and arm ID for fine-grained bimanual manipulation.

4 Method

4.1 Extending PerAct for Bimanual Manipulation

PerAct [11] was originally designed and tested for single-arm manipulation. We extend it to support bimanual manipulation. A natural way to do this would be to train separate policies for the two arms. However, we exploit the discretized action space that predicts the next best voxel with spatial equivariance properties and formulate a system that uses acting and stabilizing policies. In contrast to a policy that operates in joint-space control, acting and stabilizing policies perform the same functions irrespective of whether it is a left arm or a right arm, assuming the next best voxel is kinematically feasible for both arms. Hence, either arm can execute an acting policy or a stabilizing policy, which improves policy learning. In the low-level action space, the arms execute one low-level action a_t^s and a_t^a (see Section 3) at each time t. In the following, we use similar notation as [11] but index components as belonging to an arm using the superscript: arm $\in \{\text{acting}, \text{stabilizing}\}$.

At each time step, the input to each arm is a voxel observation $\mathbf{v}$, proprioception data of both robot arms ρ, a language goal $\mathbf{l} \in \{\ell_{as}, \ell_{sa}\}$, and an arm ID $\xi \in \{0, 1\}$, and the task is to predict an action. During training, the language goal is given in the data, but during evaluation, we use VLMs to determine which language goal, ℓ_{as} or ℓ_{sa}, to use. If the language goal is ℓ_{as}, we assign the left arm ($\xi = 0$) to the acting policy and the right arm ($\xi = 1$) to the stabilizing policy, and conversely for ℓ_{sa}. During training, this allows our method to learn to map the appropriate acting or stabilizing actions to a given arm, and during evaluation, this informs each arm's actions. Note that the predicted arm ID is discarded. PerAct uses value maps to represent different components of the action space, where predictions for each arm are $\mathcal{Q}$-functions with state-action values. Formally, we have the following five value maps *per arm*, as the output of the arm's learned deep neural network, where:

$$\begin{aligned}
\mathcal{V}^{\text{arm}}_{\text{trans}} &= \text{softmax}(\mathcal{Q}^{\text{arm}}_{\text{trans}}((x,y,z)|\mathbf{v},\rho,\mathbf{l},\xi)) \quad & \mathcal{V}^{\text{arm}}_{\text{rot}} &= \text{softmax}(\mathcal{Q}^{\text{arm}}_{\text{rot}}((\psi,\theta,\phi)|\mathbf{v},\rho,\mathbf{l},\xi)) \\
\mathcal{V}^{\text{arm}}_{\text{open}} &= \text{softmax}(\mathcal{Q}^{\text{arm}}_{\text{open}}(\omega|\mathbf{v},\rho,\mathbf{l},\xi)) & \mathcal{V}^{\text{arm}}_{\text{collide}} &= \text{softmax}(\mathcal{Q}^{\text{arm}}_{\text{collide}}(\kappa|\mathbf{v},\rho,\mathbf{l},\xi)) \quad , \\
\mathcal{V}^{\text{arm}}_{\text{id}} &= \text{softmax}(\mathcal{Q}^{\text{arm}}_{\text{id}}(\upsilon|\mathbf{v},\rho,\mathbf{l},\xi))
\end{aligned}$$

and where (x, y, z), (ψ, θ, ϕ), ω, κ, and υ represent, respectively, the 3D position, the discretized Euler angle rotations, the binary gripper opening state, the binary collision variable, and the binary arm ID. At test time, to select each arm's action, we perform an "argmax" over all the input variables to the arm's five $\mathcal{Q}$-value, to get the five components. We refer the reader to [11] for more details.

The demonstrations provide labels for each arm's five action components, giving us the following nine label sources: $Y^{\text{arm}}_{\text{trans}} \in \mathbb{R}^{L\times W\times H}$ for translations, $Y^{\text{arm}}_{\text{rot}} \in \mathbb{R}^{(360/R)\times 3}$ (with $R = 5$) for discretized rotations, $Y^{\text{arm}}_{\text{open}} \in \mathbb{R}^2$ for the binary open variables, $Y^{\text{arm}}_{\text{collide}} \in \mathbb{R}^2$ for the binary collide variables, and $Y^{\text{arm}}_{\text{id}} \in \mathbb{R}^2$ for the binary arm ID variables. The overall training loss for VoxAct-B is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{acting}} + \mathcal{L}_{\text{stabilizing}} \tag{1}$$

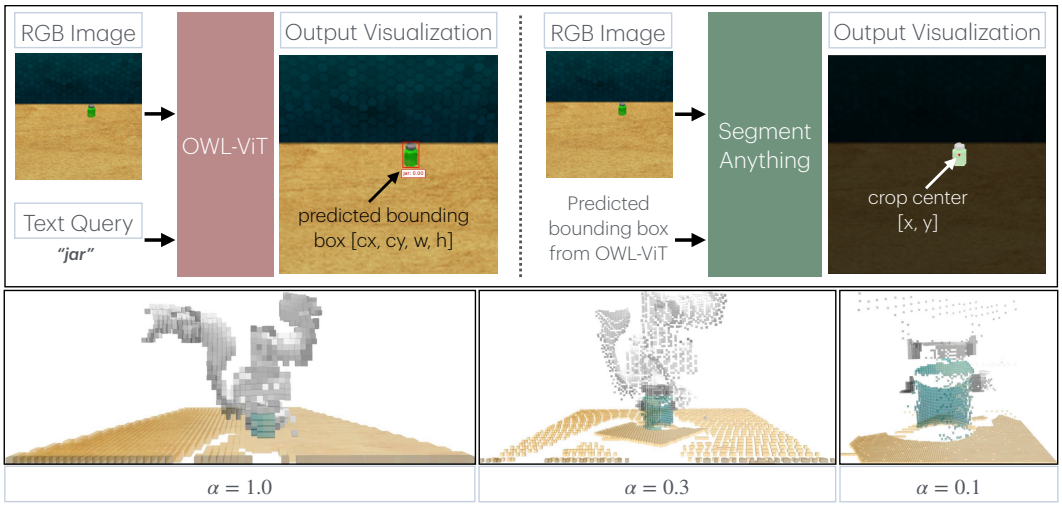

Figure 3: **Top**: VLMs usage as part of VoxAct-B, visualizing the `Open Jar` task in simulation, showing the role of OWL-ViT and Segment Anything. The RGB images from the front camera shown above are examples of actual (uncropped) images provided as input to the models. **Bottom**: visualization of different α values resulting in coarser grids ($\alpha = 1.0$) to finer grids ($\alpha = 0.1$). We use $\alpha = 0.3$ for `Open Jar`.

and where for both values of arm $\in \{\text{acting}, \text{stabilizing}\}$, we have

$$\mathcal{L}_{\text{arm}} = -\mathbb{E}_{Y^{\text{arm}}_{\text{trans}}}[\log \mathcal{V}^{\text{arm}}_{\text{trans}}] - \mathbb{E}_{Y^{\text{arm}}_{\text{rot}}}[\log \mathcal{V}^{\text{arm}}_{\text{rot}}] - \mathbb{E}_{Y^{\text{arm}}_{\text{open}}}[\log \mathcal{V}^{\text{arm}}_{\text{open}}] - \mathbb{E}_{Y^{\text{arm}}_{\text{collide}}}[\log \mathcal{V}^{\text{arm}}_{\text{collide}}] - \mathbb{E}_{Y^{\text{arm}}_{\text{id}}}[\log \mathcal{V}^{\text{arm}}_{\text{id}}], \tag{2}$$

which consists of a set of cross-entropy classifier-style losses for each component in the action.

4.2 VoxAct-B: Voxel Representations and PerAct for Bimanual Manipulation

When using voxel representations for fine-grained manipulation, a high voxel resolution is essential. While one can increase the number of voxels, this would consume more memory, slow down training, and adversely affect learning as the policy is optimizing over a larger state space. Therefore, given a voxel grid observational input $\mathbf{v}$ of size $(L \times W \times H)$ that spans x^3 meters of the workspace, we keep the number of voxels the same but reduce the relevant workspace. We use VLMs to detect the object of interest in the scene and "crop" the grid around this object, resulting in a voxel grid that spans αx^3 meters of the workspace, where α is a fraction that determines the size of the crop. This allows zooming into the more important region of interest. The voxel resolution becomes $(\frac{L}{\alpha x}, \frac{W}{\alpha x}, \frac{H}{\alpha x})$ voxels/meters from the original resolution of $(\frac{L}{x}, \frac{W}{x}, \frac{H}{x})$ voxels/meters.

To detect the object of interest reliably, we use a two-stage approach similar to [16]. We input a text query to an open-vocabulary object detector to detect the object. Then, we use a foundational image segmentation model to obtain the segmentation mask of the object and use the mask's centroid along with point cloud data to retrieve the object's pose with respect to the front camera. We use the pose of the object to determine the task-specific roles of each arm and the language goal. This cropped voxel grid and language goal are the input to our bimanual manipulation policy. We call our method *VoxAct-B: Voxel-Based Acting and Stabilizing Policy*. See Figures 2 and 3 for an overview.

4.3 Additional Implementation Details

The bimanual manipulation policy uses a voxel grid size of 50^3 that spans 2^3 meters. The proprioception data includes: the gripper opening state of both arms, the positions of the left arm left finger, left arm right finger, right arm left finger, right arm right finger, and timestep. Following PerAct [11], we apply data augmentations to the training data using SE(3) transformations: $[\pm 0.125\,\text{m}, \pm 0.125\,\text{m}, \pm 0.125\,\text{m}]$ in translations and $\pm 45°$ in the yaw axis. We use 2048 latents of dimension 512 in the Perceiver Transformer [62] and optimize the entire network using the LAMB [63] optimizer. We use $\alpha = 0.3$ for `Open Jar` and $\alpha = 0.4$ for the drawer tasks. We

select these α values by using a starting state of the environment with the largest scaling size factor for the object of interest and checking whether the object remains entirely contained in the voxel grid after cropping. We train the policy with a batch size of 1 on a single Nvidia 3000 series GPU for two days. For VLMs, we use OWL-ViT [64] as our open-vocabulary object detection algorithm and Segment Anything [65] as our foundational image segmentation model.

5 Experiments

5.1 Tasks

In simulation, we build on top of RLBench [14], a popular robot manipulation benchmark widely used in prior work, including VoxPoser and PerAct. We extend it to support bimanual manipulation (see Appendix A for details). We do not perform simulation-to-real transfer in this paper; simulation is for algorithm development and benchmarking. We design the following three bimanual tasks:

- **`Open Jar`**: a jar with a screw-on lid is randomly spawned and scaled from 90% to 100% of the original size within the robot's 0.43×0.48 meters of workspace. The jar color is uniformly sampled from a set of 20 colors. The robot must first grasp the jar with one hand and use the other to unscrew the lid in an anti-clockwise direction until it is completely removed.
- **`Open Drawer`**: a drawer is randomly spawned inside a workspace of 0.65×0.91 meters. It is randomly scaled from 90% to 100% of its original size, and its rotation is randomized between $-\frac{\pi}{8}$ and $\frac{\pi}{8}$ radians. The robot needs to stabilize the top of the drawer with one hand and then open the bottom drawer with the other.
- **`Put Item in Drawer`**: a drawer (the same type from `Open Drawer`) is randomly spawned in a workspace of 0.65×0.91 meters, and is randomly scaled and rotated using the same sampling ranges from `Open Drawer`. The robot needs to open the top drawer with one hand, grasp the item placed on top of the drawer with the other hand, and place it in the top drawer.

See Figure 1 for an illustration. In the real world, we test `Open Jar` and `Open Drawer` using a coffee jar with dimensions $3.35 \times 2.85 \times 4.8$ inches and a drawer of dimensions $12 \times 12 \times 12$ inches.

5.2 Baselines and Ablations

In simulation, we compare against several strong baseline methods: **Action Chunking with Transformers (ACT)** [3], **Diffusion Policy** [15], and **VoxPoser** [16]. ACT is a state-of-the-art method for bimanual manipulation. Diffusion Policy represents the policy as a conditional denoising diffusion process and excels at learning multimodal distributions. ACT and Diffusion Policy use joint positions for their action space instead of predicting end-effector poses as our method. We adapt the Mobile ALOHA repository for ACT and a CNN-based Diffusion Policy, and we tune their parameters (e.g., chunk size and action horizon) to improve performance. For VoxPoser, we write and tune their LLM prompts to work on our bimanual manipulation tasks using the VoxPoser repository. Additionally, we include a **Bimanual PerActs** baseline, which trains separate PerAct policies for the left and right arms, to show how a straightforward bimanual adaptation of a single-arm, state-of-the-art voxel-based method performs. It uses the same number of voxels, 100^3, as the original PerAct. See the Appendix for further details. We also test the following ablations of VoxAct-B:

- **VoxAct-B w/o VLMs**: does not use the VLMs to detect the object of interest and crop the voxel grid. It uses the same number of voxels as our method and the default workspace dimensions.
- **VoxAct-B w/o acting and stabilizing formulation**: trains a left-armed policy for left arm actions and a right-armed policy for right arm actions. Otherwise, it is the same as VoxAct-B.
- **VoxAct-B w/o arm ID**: excludes arm ID as input and disables the corresponding loss function.

5.3 Experiment Protocol and Evaluation

To generate demonstrations in simulation, we follow the convention from RLBench and define a sequence of waypoints to complete the task, and use motion planning to control the robot arms to

reach waypoints. We generate 10 and 100 demonstrations of training data. Half of this data consists of left-acting and right-stabilizing demonstrations, and the other half contains right-acting and left-stabilizing demonstrations. We generate 25 episodes of validation and test data using different random seeds. We train and evaluate all methods using three random seeds and report the average of the results. We evaluate all methods on the same set of test demonstrations for a fair comparison.

Each method saves a checkpoint every 10,000 training steps. For all methods, we use the best-performing checkpoint, evaluated on the validation data, to obtain the test success rate and report this result. Deciding the best checkpoints for VoxAct-B and ablations is nontrivial since iterating over all possible combinations is computationally expensive. For example, with 400,000 training steps, using the same 10,000 checkpoint interval means there are $40 \times 40 = 1600$ possible combinations. Therefore, with the validation data, we use the latest stabilizing checkpoint to evaluate all acting checkpoints; we use the best acting checkpoint to evaluate all stabilizing checkpoints. Then, we use the best-performing acting and stabilizing checkpoints to obtain the test success rate.

In the real world, we use a dual-arm CB2 UR5 robot setup. Each arm has 6-DOFs and has a Robotiq 2F-85 parallel-jaw gripper. We collect ten demonstrations for each task with the GELLO teleoperation interface [66]. We use a flat workspace with dimension 0.97 m by 0.79 m and mount an Intel RealSense D415 RGBD camera at a height of 0.42 m at a pose which reduces occlusions of the object. For evaluation, we perform 10 consecutive rollouts to record the results for each task. In `Open Drawer`, the arms have fixed roles of right acting and left stabilizing, and the acting arm opens the top drawer. The drawer has variations of 10 cm in translations and 20° of rotations. In `Open Jar`, the roles of the arms are reversed and fixed, and the jar has variations of 12 cm in translations. See Appendix B for more details.

6 Results

6.1 Simulation Results

Comparisons with baselines. Table 1 reports the test success rates of baselines and VoxAct-B. When we train all methods using ten demonstrations, VoxAct-B outperforms all baselines by a large margin. In a low-data regime, the discretized action space with spatial equivariance properties (as used in VoxAct-B and Bimanual PerActs) may be more sample-efficient and easier for learning-based methods compared to methods that use joint space (ACT and Diffusion Policy). When we train all methods using more demonstrations (100), VoxAct-B still outperforms baselines on most tasks, except ACT for `Put Item in Drawer`. Through ablations of ACT and Diffusion Policy, we found that fixing the roles of acting and stabilizing arms greatly improved their performance. We theorize that they lack a grounding mechanism that allows both arms to learn acting and stabilizing actions effectively. We attribute the tasks' difficulty to the following: high environment variation, difficult bimanual manipulation tasks with high-dimensional action spaces and fine-grained manipulation, and the two types of training data that each method needs to learn (based on which arms are acting and stabilizing).

	Open Jar		Open Drawer		Put Item in Drawer	
Method	10	100	10	100	10	100
Diffusion Policy	5.3	24.0	4.0	6.7	2.7	6.7
ACT w/Transformers	1.3	26.7	10.7	29.3	8.0	**50.7**
VoxPoser	8.0	8.0	32.0	32.0	4.0	4.0
Bimanual PerActs	9.3	-	40.0	-	6.7	-
VoxAct-B (ours)	**38.6**	**58.7**	**73.3**	**73.3**	**36.0**	46.7

Table 1: Performance of different methods on bimanual manipulation tasks in simulation, based on 10 or 100 (task-specific) training demonstrations. We use three training seeds for all methods, and evaluate on the same 25 episodes of unseen test data using the best checkpoints from validation (Section 5.3). The results are the average evaluation over three seeds. We only test Bimanual Peracts with ten demonstrations (not 100) due to computational constraints. VoxPoser does not have training, so its 10 and 100 results are identical.

Qualitatively, baseline methods, especially VoxPoser, typically struggle with precisely grasping objects such as drawer handles and jars. The baselines also struggle with correctly assigning the roles of each arm. For instance, a policy intended to execute acting actions may unpredictably produce stabilizing actions. Furthermore, they can generate kinematically infeasible actions or actions that

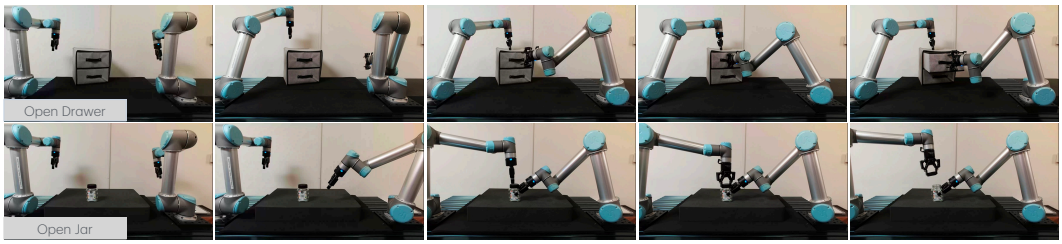

Figure 4: Example successful rollouts (one per row) of VoxAct-B on a real-world bimanual setup with UR5s.

lead to erratic movements, as seen in ACT and Diffusion Policy, which may be caused by insufficient training data. In contrast, we observe fewer of these errors with VoxAct-B.

Ablation experiments. Table 2 reports results on `Open Drawer` in simulation, based on 10 training demonstrations. We use the same training and evaluation protocols as Table 1. VoxAct-B w/o VLMs performs poorly versus VoxAct-B because, without using the VLMs to reduce the physical space, the voxel resolution is lower due to the large workspace area for each individual voxel, which hinders fine-grained manipulation. Moreover, VoxAct-B w/o acting and stabilizing and VoxAct-B w/o arm ID perform worse than VoxAct-B, and they also struggle with the same issues as the baselines.

Method	Open Drawer
VoxAct-B w/o VLMs	16.0
VoxAct-B w/o acting and stabilizing	66.7
VoxAct-B w/o arm ID	68.0
VoxAct-B (ours)	**73.3**

Table 2: Ablation experiment results in simulation.

6.2 Physical Results

Figure 4 shows real-world examples of VoxAct-B. In `Open Drawer`, success is when the stabilizing arm holds the drawer from the top while the acting arm pulls the top part. VoxAct-B succeeds in 6 out of 10 trials; the failures include robot joints hitting their limits, imprecision in grasping the handle, and collisions with the drawer. In `Open Jar`, a success is when the stabilizing arm grasps the jar while the acting arm unscrews the lid. VoxAct-B succeeds in 5 out of 10 trials. While the stabilizing arm performs well in grasping the jar (9 out of 10 successes), the acting arm struggles with unscrewing the lid, succeeding only 5 out of 10 times due to imprecise grasping of the lid.

6.3 Limitations and Failure Cases

VoxAct-B implicitly assumes the object of interest does not encompass most of the workspace. If it does, it will be difficult to crop the voxel grid without losing relevant information. Another limitation is that VoxAct-B depends on the quality of VLMs. We have observed that some failures come from poor detection and segmentation from VLMs, which causes VoxAct-B to output undesirable actions. In addition to common errors described in Section 6.1, for `Put Item in Drawer`, VoxAct-B tends to struggle more with executing acting actions (e.g., drawer-opening and cube-picking/placing actions) in contrast to stabilizing actions.

7 Conclusion

In this paper, we present VoxAct-B, a voxel-based, language-conditioned method for bimanual manipulation. We use VLMs to focus on the most important regions in the scene and reconstruct a voxel grid around them. This approach enables the policy to process the same number of voxels within a reduced physical space, resulting in a higher voxel resolution necessary for accurate, fine-grained bimanual manipulation. VoxAct-B outperforms strong baselines, such as ACT, Diffusion Policy, and VoxPoser, by a large margin on difficult bimanual manipulation tasks. We also demonstrate VoxAct-B on real-world `Open Drawer` and `Open Jar` tasks using a dual-arm UR5 robot. We hope that this inspires future work in asymmetric bimanual manipulation tasks.

References

[1] F. Krebs and T. Asfour. A Bimanual Manipulation Taxonomy. In *IEEE Robotics and Automation Letters (RA-L)*, 2022.

[2] R. Chitnis, S. Tulsiani, S. Gupta, and A. Gupta. Efficient bimanual manipulation using learned task schemas. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[3] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems (RSS)*, 2023.

[4] G. Franzese, L. d. S. Rosa, T. Verburg, L. Peternel, and J. Kober. Interactive imitation learning of bimanual movement primitives. *IEEE/ASME Transactions on Mechatronics*, 28(1):1–13, 2023.

[5] Y. Avigal, L. Berscheid, T. Asfour, T. Kröger, and K. Goldberg. SpeedFolding: Learning Efficient Bimanual Folding of Garments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[6] J. Grannen, P. Sundaresan, B. Thananjeyan, J. Ichnowski, A. Balakrishna, V. Viswanath, M. Laskey, J. E. Gonzalez, and K. Goldberg. Untangling dense knots by learning task-relevant keypoints. In *Conference on Robot Learning (CoRL)*, 2020.

[7] F. Xie, A. Chowdhury, M. C. De Paolis Kaluza, L. Zhao, L. L. Wong, and R. Yu. Deep imitation learning for bimanual robotic manipulation. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[8] C. Bersch, B. Pitzer, and S. Kammel. Bimanual robotic cloth manipulation for laundry folding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[9] J. Grannen, Y. Wu, S. Belkhale, and D. Sadigh. Learning Bimanual Scooping Policies for Food Acquisition. In *Conference on Robot Learning (CoRL)*, 2022.

[10] H. Ha and S. Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *Conference on Robot Learning (CoRL)*, 2021.

[11] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

[12] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox. RVT: Robotic View Transformer for 3D Object Manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[13] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki. Act3D: 3D Feature Field Transformers for Multi-Task Robotic Manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[14] S. James, Z. Ma, D. Rovick Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[15] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[16] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning (CoRL)*, 2023.

[17] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel. Cloth Grasp Point Detection Based on Multiple-View Geometric Cues with Application to Robotic Towel Folding. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[18] A. Canberk, C. Chi, H. Ha, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Cloth funnels: Canonicalized-alignment for multi-purpose garment manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[19] A. Colomé and C. Torras. Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes. In *IEEE Transactions on Robotics*, 2018.

[20] G. Salhotra, I.-C. A. Liu, and G. Sukhatme. Learning robot manipulation from cross-morphology demonstration. In *Conference on Robot Learning (CoRL)*, 2023.

[21] T. Weng, S. Bajracharya, Y. Wang, K. Agrawal, and D. Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning (CoRL)*, 2021.

[22] L. Y. Chen, B. Shi, D. Seita, R. Cheng, T. Kollar, D. Held, K. Goldberg, K. Goldberg, K. Goldberg, K. Goldberg, and K. Goldberg. AutoBag: Learning to Open Plastic Bags and Insert Objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[23] L. Y. Chen, B. Shi, R. Lin, D. Seita, A. Ahmad, R. Cheng, T. Kollar, D. Held, and K. Goldberg. Bagging by Learning to Singulate Layers Using Interactive Perception. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[24] A. Bahety, S. Jain, H. Ha, N. Hager, B. Burchfiel, E. Cousineau, S. Feng, and S. Song. Bag All You Need: Learning a Generalizable Bagging Strategy for Heterogeneous Objects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[25] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang. Dynamic handover: Throw and catch with bimanual hands. In *Conference on Robot Learning (CoRL)*, 2023.

[26] L. Yan, T. Stouraitis, J. Moura, W. Xu, M. Gienger, and S. Vijayakumar. Impact-Aware Bimanual Catching of Large-Momentum Objects. In *IEEE Transactions on Robotics*, 2024.

[27] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik. Twisting Lids Off with Two Hands. *arXiv preprint arXiv:2403.02338*, 2024.

[28] Y. Chen, Y. Yang, T. Wu, S. Wang, X. Feng, J. Jiang, S. M. McAleer, H. Dong, Z. Lu, and S.-C. Zhu. Towards Human-Level Bimanual Dexterous Manipulation with Reinforcement Learning. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[29] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning Visuotactile Skills with Two Multifingered Hands. *arXiv preprint arXiv:2404.16823*, 2024.

[30] K. Zakka, P. Wu, L. Smith, N. Gileadi, T. Howell, X. B. Peng, S. Singh, Y. Tassa, P. Florence, A. Zeng, and P. Abbeel. Robopianist: Dexterous piano playing with deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2023.

[31] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. DexCap: Scalable and Portable Mocap Data Collection System for Dexterous Manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[32] J. Yang, C. Deng, J. Wu, R. Antonova, L. Guibas, and J. Bohg. EquivAct: SIM(3)-Equivariant Visuomotor Policies beyond Rigid Object Manipulation. *arXiv preprint arXiv:2310.16050*, 2023.

[33] K. Chu, X. Zhao, C. Weber, M. Li, W. Lu, and S. Wermter. Large Language Models for Orchestrating Bimanual Robots. *arXiv preprint arXiv:2404.02018*, 2024.

[34] S. Stepputtis, M. Bandari, S. Schaal, and H. Ben Amor. A System for Imitation Learning of Contact-Rich Bimanual Manipulation Policies. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[35] A. Bahety, P. Mandikal, B. Abbatematteo, and R. Martín-Martín. ScrewMimic: Bimanual Imitation from Human Videos with Screw Space Projection. In *Robotics: Science and Systems (RSS)*, 2024.

[36] L. X. Shi, A. Sharma, T. Z. Zhao, and C. Finn. Waypoint-based imitation learning for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[37] Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *arXiv preprint arXiv:2401.02117*, 2024.

[38] A. . Team. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation, 2024. URL https://aloha-2.github.io/.

[39] J. Grannen, Y. Wu, B. Vu, and D. Sadigh. Stabilize to act: Learning to coordinate for bimanual manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[40] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo Grau, N. Fazeli, F. Alet, N. Dafle, R. Holladay, I. Morena, P. Nair, D. Green, I. Taylor, W. Liu, and A. Rodriguez. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[41] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser. Learning Synergies between Pushing and Grasping with Self-supervised Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[42] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2020.

[43] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng. Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[44] R. Lee, D. Ward, A. Cosgun, V. Dasagi, P. Corke, and J. Leitner. Learning arbitrary-goal fabric folding with one hour of real robot experience. In *Conference on Robot Learning (CoRL)*, 2020.

[45] M. Shridhar, L. Manuelli, and D. Fox. CLIPort: What and Where Pathways for Robotic Manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[46] S. James, K. Wada, T. Laidlow, and A. J. Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[47] OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

[48] H. Touvron and Others. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023.

[49] G. T. Google. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*, 2023.

[50] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents. In *International Conference on Machine Learning (ICML)*, 2022.

[51] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. PaLM-E: An Embodied Multimodal Language Model. In *International Conference on Machine Learning (ICML)*, 2023.

[52] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning (CoRL)*, 2022.

[53] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su. LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[54] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[55] W. Huang, F. Xia, D. Shah, D. Driess, A. Zeng, Y. Lu, P. R. Florence, I. Mordatch, S. Levine, K. Hausman, and B. Ichter. Grounded decoding: Guiding text generation with grounded models for robot control. *arXiv preprint arXiv:2303.00855*, 2023.

[56] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. ProgPrompt: Program generation for situated robot task planning using large language models. *Autonomous Robots (AURO)*, 2023.

[57] K. Kawaharazuka, T. Matsushima, A. Gambardella, J. Guo, C. Paxton, and A. Zeng. Real-World Robot Applications of Foundation Models: A Review. *arXiv preprint arXiv:2402.05741*, 2024.

[58] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman, B. Ichter, D. Driess, J. Wu, C. Lu, and M. Schwager. Foundation Models in Robotics: Applications, Challenges, and the Future. *arXiv preprint arXiv:2312.07843*, 2023.

[59] Y. Hu, Q. Xie, V. Jain, J. Francis, J. Patrikar, N. Keetha, S. Kim, Y. Xie, T. Zhang, S. Zhao, Y. Q. Chong, C. Wang, K. Sycara, M. Johnson-Roberson, D. Batra, X. Wang, S. Scherer, Z. Kira, F. Xia, and Y. Bisk. Toward General-Purpose Robots via Foundation Models: A Survey and Meta-Analysis. *arXiv preprint arXiv:2312.08782*, 2023.

[60] J. Varley, S. Singh, D. Jain, K. Choromanski, A. Zeng, S. B. R. Chowdhury, A. Dubey, and V. Sindhwani. Embodied AI with Two Arms: Zero-shot Learning, Safety and Modularity. *arXiv preprint arXiv:2404.03570*, 2024.

[61] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1760–1770, 2021. doi:10.1109/ICCV48922.2021.00180.

[62] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, A. Brock, E. Shelhamer, O. J. H'enaff, M. M. Botvinick, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver io: A general architecture for structured inputs & outputs. *ArXiv*, abs/2107.14795, 2021. URL https://api.semanticscholar.org/CorpusID:236635379.

[63] Y. You, J. Li, S. J. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv: Learning*, 2019. URL https://api.semanticscholar.org/CorpusID:165163737.

[64] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, et al. Simple open-vocabulary object detection. In *European Conference on Computer Vision*, pages 728–755. Springer, 2022.

[65] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. B. Girshick. Segment anything. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3992–4003, 2023. URL https://api.semanticscholar.org/CorpusID:257952310.

[66] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators, 2023.

[67] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3D Diffuser Actor: Policy Diffusion with 3D Scene Representations. *arXiv preprint arXiv:2402.10885*, 2024.

[68] S. Chen, R. Garcia, C. Schmid, and I. Laptev. PolarNet: 3D Point Clouds for Language-Guided Robotic Manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

[69] P.-L. Guhur, S. Chen, R. Garcia, M. Tapaswi, I. Laptev, and C. Schmid. Instruction-driven History-aware Policies for Robotic Manipulations. In *Conference on Robot Learning (CoRL)*, 2022.

[70] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su. Maniskill2: A unified benchmark for generalizable manipulation skills. In *International Conference on Learning Representations (ICLR)*, 2023.