IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

BOSS: Benchmark for Observation Space Shift in Long-Horizon Task

Yue Yang[1], Linfeng Zhao[2], Mingyu Ding[1], Gedas Bertasius[1], and Daniel Szafir[1]

***Abstract*—Robotics has long sought to develop robots capable of completing previously unseen long-horizon tasks. Hierarchical approaches offer a pathway for achieving this goal by executing skill combinations arranged by a task planner, with each visuomotor skill pre-trained using a specific imitation learning (IL) algorithm. However, even in simple long-horizon tasks like skill chaining, hierarchical approaches often struggle due to a problem we identify as Observation Space Shift (OSS), where the sequential execution of preceding skills causes shifts in the observation space, disrupting the performance of subsequent individually trained skill policies. To understand OSS and evaluate its impact on long-horizon tasks, we introduce BOSS (a Benchmark for Observation Space Shift). BOSS comprises three distinct challenges: "Single Predicate Shift", "Accumulated Predicate Shift", and "Skill Chaining", each designed to assess a different aspect of OSS's negative effect. We evaluated several recent popular IL algorithms on BOSS, including three Behavioral Cloning methods and the Visual Language Action model OpenVLA. Even on the simplest challenge, we observed average performance drops of 67%, 35%, 34%, and 54%, respectively, when comparing skill performance with and without OSS. Additionally, we investigate three potential solutions, including using frozen robotics-specific vision encoders, switching to 3D pointcloud-based inputs, and applying data augmentation to expand visual diversity. Our results show that none of these approaches are sufficient to resolve OSS. The project page is: `https://boss-benchmark.github.io/`**

***Index Terms*—Imitation Learning, Long-Horizon Task, Robotics**

I. Introduction

RECENT advances in robot learning have shown promise in diverse manipulation applications, including manufacturing [1], sports [2], [3], and household tasks [4], [5]. Imitation Learning (IL), which empowers end users to teach robot skills and behaviors through demonstrations, has become a prevalent approach in developing various skill controllers [6], [7]. Skill chaining, the sequential execution of such pre-learned skills, is a simple yet powerful structure for tackling complex long-horizon robot tasks [8], [9]. However, failures often arise in a skill chain when a skill encounters initial states that were not seen during training [10], [11]. Specifically, the terminal state of a preceding skill may fall outside the initial state distribution the next skill policy was trained on, leading to task failure.

As an example, consider a household long-horizon task illustrated in Fig. 1, which involves two skills: `PlaceObject(potato, bowl)` and `MoveContainer(bowl, cabinet)`. When training the visuomotor policy for `MoveContainer(bowl, cabinet)`, the initial state set only includes visual states (i.e., images) with an empty bowl. However, the terminal state of `PlaceObject(potato, bowl)` places a potato in the bowl, creating a visual mismatch that can cause the `MoveContainer(bowl, cabinet)` policy to fail. We refer to this issue as **Observation Space Shift (OSS)**, a problem that frequently arises during skill transitions in visual-input long-horizon tasks (§III formally defines OSS).

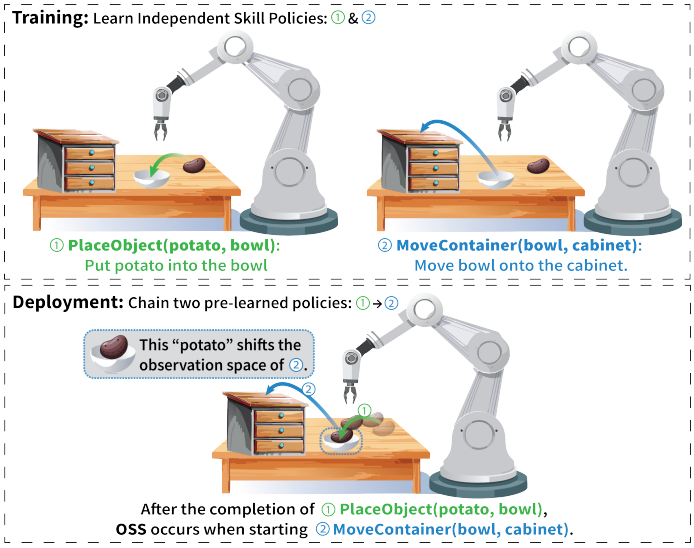

Fig. 1. The example illustrates how Observation Space Shift (OSS) occurs when chaining two pre-trained skills: `PlaceObject(potato, bowl)` and `MoveContainer(bowl, cabinet)`. During deployment, OSS arises in `MoveContainer(bowl, cabinet)` because the observation space changes, with a potato inside the bowl instead of the empty bowl scenario from `MoveContainer(bowl, cabinet)`'s pre-training.

As a result, it is critical to ensure smooth skill transitions in skill chaining for long-horizon tasks [12]. Previous researchers tackle the skill transition problem through two main strategies. Offline approaches involve learning a transition policy to shift the state from the terminal state of one skill to the initial state of the next [10], [13]–[15]. Alternatively, a second type of approach ensures that the initial state set of the next skill matches the terminal state set of the previous skill by fine-tuning the policy of the preceding skill, the next skill, or both in an online manner [12], [16]–[19]. However, all of these previous works rely on assumptions that are frequently impractical for visuomotor IL policies. Offline methods assume that the initial states for each skill are always reachable, which is often unreasonable for visual-input long-horizon tasks,

Manuscript received: February 16, 2025; Revised: May 14, 2025; Accepted: June 8, 2025.

This paper was recommended for publication by Editor Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by NSF Award 2222953, the Laboratory for Analytic Sciences via NC State University, and ONR Award N00014-23-1-2356.

[1] The University of North Carolina at Chapel Hill. `yygx@cs.unc.edu`
[2] Northeastern University.

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

as it may require undoing previously completed skills. For example, as shown in Figure 1, offline methods would need to learn a transition policy to remove the potato from the bowl in order to enable the observation space to match the initial state of `MoveContainer(bowl, cabinet)` (i.e., an empty bowl), which is logically invalid. Online approaches, on the other hand, assume that policies can be continuously fine-tuned to handle new initial states. This assumption is equally problematic for visual-input long-horizon tasks, where OSS is highly likely to happen at each skill transition. The frequent occurrence of OSS would necessitate repeated fine-tuning, leading to substantial computational overhead and inefficiency.

These limitations motivate us to propose a dedicated **Benchmark for Observation Space Shift (BOSS)** to demonstrate the significant negative impact OSS has on long-horizon tasks, facilitate a deeper understanding of the problem, and inspire potential solutions (§IV). Built on the LIBERO simulator [20], the benchmark consists of evaluating three progressive challenges: 1) *BOSS-C1* evaluates the robustness of skill policies against a single observation modification caused by the preceding skill. For example, as shown in Figure 1, we evaluate the robustness of the `MoveContainer(bowl, cabinet)` policy under a single modification (e.g., a potato placed in the bowl). 2) *BOSS-C2* builds on this by evaluating the cumulative negative effects of multiple modifications introduced by several preceding skills, creating a more complex and challenging scenario than BOSS-C1. For instance, in the example from Figure 1, adding a new skill `OpenDrawer(cabinet)` before `PlaceObject(potato, bowl)` results in accumulated modifications for the `MoveContainer(bowl, cabinet)` policy (i.e., the drawer is open, and a potato is placed in the bowl). 3) *BOSS-C3* focuses on a realistic long-horizon task comprising a three-skill chain, testing policy performance across the entire sequence (e.g., `OpenDrawer(cabinet)` $\rightarrow$ `PlaceObject(potato, bowl)` $\rightarrow$ `MoveContainer(bowl, cabinet)`).

We assess four popular IL algorithms using these benchmarks, offering detailed analyses of the results to generate insights and set baselines for future research (§V). In addition, we examine three intuitive strategies to mitigate the OSS problem: using frozen robotics-specific vision encoders such as R3M [21] and LIV [22], switching to 3D pointcloud-based policies such as 3D Diffuser Actor [23], and expanding the Libero dataset through data augmentation with our proposed Rule-based Automatic Modification Generator (RAMG). While these strategies aim to improve robustness by leveraging pretrained representations, alternative input modalities, or larger and more diverse datasets, our comprehensive experiments show that none of them are sufficient to resolve OSS. These findings provide key insights and motivate the need for more targeted algorithmic solutions (§VI). In summary, our contributions of this work are three-fold:

1) For the first time, we formulate the **Observation Space Shift (OSS)**, a critical problem in long-horizon robotic tasks.
2) We introduce BOSS, a comprehensive benchmark that evaluates four IL methods across three increasingly challenging scenarios of OSS in long-horizon manipulation.
3) We explore three intuitive strategies to mitigate OSS, including using frozen robotics-specific vision encoders, switching to 3D pointcloud-based inputs, and applying data augmentation using a large and diverse dataset generated by our Rule-based Automatic Modification Generator. Our results show that none of these strategies are sufficient to resolve OSS, emphasizing the need for new algorithmic solutions in future research.

II. Related Work

A. Robot Learning Benchmarks

In recent years, numerous benchmarks for robot learning research have been developed. For example, Meta-World [24] targets meta-reinforcement learning and multi-task learning. RLBench [25] provides 100 simulated household tasks. D4RL [26] focuses on offline reinforcement learning across diverse tasks. Robomimic [27] emphasizes learning from human demonstrations. ManiSkill2 [28] supports a wide range of tasks incorporating both proprioceptive and visual data. However, none of these benchmarks are designed explicitly for long-horizon tasks. Some recent benchmarks also evaluate policy robustness under changes in the observation space, but these variations are externally introduced, rather than arising from prior skill executions, and thus are likewise not tailored for long-horizon tasks [29], [30].

On the other hand, CausalWorld [31] includes long-horizon tasks but focuses on causal structures and transfer learning. Calvin [32] addresses language-conditioned multi-task policy learning. FurnitureBench [33] features realistic, long-horizon furniture assembly tasks. RoboCasa [34] offers a variety of long-horizon tasks set in kitchen environments, created using generative models. LoHoRavens [35] is tailored for language-conditioned long-horizon tasks. Despite their significant contributions, these benchmarks do not focus on skill transitions and are thus not well-suited for studying the OSS problem, which demands diverse skill transitions within long-horizon tasks. Libero [20], while primarily designed for lifelong robot learning, provides diverse tasks and customization capabilities, making it an ideal foundation for developing our BOSS benchmark, which provides diverse and large-scale skill transitions, and specifically targets the OSS problem.

B. Skill Chaining

Skill chaining, a fundamental structure in HIL, sequentially executing pre-learned skill policies to complete long-horizon tasks [9], [36]–[40], where ensuring seamless transitions between adjacent skills is the primary challenge. One approach addresses this by learning transition policies to bridge the gap between the terminal state of the previous skill and the initial state of the next [10], [13]–[15]. Lee et al., 2019 proposed transition policies with proximity predictors to connect primitive skills for sequential skills [10], while adversarial inverse reinforcement learning is used to learn transition policies by matching state and action distributions [14]. Goal-conditioned policies have also been employed to imitate transitions from target demonstrations [13]. For example, Dalal et al., 2024 proposed estimating the next skill's initial pose and applying

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

motion planning to generate transitions between skills [15]. Another approach focuses on enforcing alignment between the terminal state set of the previous skill and the initial state set of the next. Expanding the initial state set of the next skill to include the terminal state set of the previous one is one method [11], though it can lead to significant state space growth over time. Lee et al., 2021 address this issue by using an adversarial learning framework to regularize the terminal state distribution of the previous skill to match the initial state distribution of the next skill [17]. Additionally, contrastive learning has been applied to learn auxiliary rewards that guide terminal states closer to the initial states of the next skill [12]. Chen et al., 2023, optimized entire policy chains to ensure dynamic transition feasibility in dexterous tasks [16]. More recently, given a fixed skill set, offline reinforcement learning has been used to generate new skills whose terminal states align with the initial states of next skills selected with LLM assistance [18], [19]. Despite these efforts, all methods rely on the two strong assumptions outlined in §I: the reachability of initial states for each skill and the feasibility of continuous policy fine-tuning, which substantially limit their ability to address the OSS problem effectively in general.

III. Problem Definition

We focus on long-horizon tasks that chain skill policies trained through imitation learning using visual inputs, assuming a robust planner generates the skill sequence based on a high-level task goal.

We first model the environment as a Partially Observable Markov Decision Process (POMDP), defined as $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{Z}, r, \gamma \rangle$. Here, $\mathcal{S}$, $\mathcal{O}$, and $\mathcal{A}$ represent the state, observation, and action spaces, respectively. The transition and observation probabilities are $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ and $\mathcal{Z}(o_t \mid s_t)$. The reward function $r(s_t, a_t, s_{t+1})$ and discount factor γ guide the trade-off between immediate and future rewards. In our setup, the observation space $\mathcal{O}$ includes both third-person camera views and proprioceptive states. The objective is, for each skill, to learn a policy $\pi : \mathcal{O} \to \Delta(\mathcal{A})$ that replicates demonstrated behavior from observation-action pairs $\tau = (o_0, a_0, \cdots, o_t, a_t)$, without explicit access to $\mathcal{S}$.

A significant problem arises during deployment when chaining skill policies, which we define as the **Observation Space Shift (OSS)**. We adopt the framework of Task and Motion Planning (TAMP) to formalize this problem [36], [37]. We define predicates $\Psi = \{\psi_1, \psi_2, \ldots, \psi_n\}$ as properties or relationships within the state space, where each predicate ψ_i maps a state $s \in \mathcal{S}$ to a truth value, $\psi_i(s) \in \{0, 1\}$. Essentially, each predicate acts as a binary classifier over the state space, identifying a subset of $\mathcal{S}$ where the predicate evaluates to true. As the example shown in Figure 1, the predicate `In(potato, bowl)` evaluates to 1 if the potato is inside the bowl, and 0 otherwise. Operator is defined as $\texttt{op} = \langle \texttt{Pre}, \texttt{Eff}, c \rangle$ that comprises three components: preconditions $\texttt{Pre} \subseteq \Psi$, effects $\texttt{Eff} \subseteq \Psi$, and a cost c. Intuitively, an operator specifies when a skill can execute (i.e., `Pre`) and what is expected after execution (i.e., `Eff`). Symbolic operators use preconditions (e.g., `On(bowl, table)`) and effects (e.g., `On(bowl, cabinet)`) to represent only the elements (e.g., bowl instead of potato) necessary for feasibility of a skill (e.g., `MoveContainer(bowl, cabinet)`), abstracting away irrelevant ones (e.g., `In(potato, bowl)`). Formally, a predicate $\psi_i \in \Psi$ is irrelevant for an operator `op` if $\psi_i \notin \texttt{Pre} \cup \texttt{Eff}$, meaning it does not influence the feasibility or expected outcome of the skill execution. However, these skill-irrelevant elements often appear in visual observations and can be unintentionally modified by preceding skills' effects. Such changes disrupt visuomotor policies, causing failures in skill execution. This problem, where changes in irrelevant predicates within the visual observation space $\mathcal{O}$ hinder visuomotor performance, is defined as OSS.

IV. The BOSS Benchmark

We introduce the BOSS benchmark, designed to facilitate a comprehensive empirical study of the OSS problem across modern IL methods.

A. Environment

We build the environment of BOSS on the Libero platform [20], which, although originally designed for lifelong robot learning tasks, provides diverse manipulation scenes featuring a Franka Emika Panda robot arm and accommodates a wide variety of tasks. Libero is highly flexible for creating and customizing tasks because they are generated using Planning Domain Definition Language (PDDL) files, which specify the planning problem including operators, predicates, and their relationships. These features make Libero an excellent foundation for developing BOSS, enabling us to adapt it specifically to study the OSS problem.

B. Task Design

Building on the Libero environment, BOSS focuses on studying the impact of OSS on long-horizon tasks. To achieve this, we require a set of atomic robotic tasks, where each task involves only a single skill. These tasks are used to simulate individual skills within a skill chain that are unaffected by OSS. Additionally, we generate modified counterparts to simulate scenarios where OSS occurs. This setup enables performance comparisons between the two sets when evaluating IL methods. All three challenges are based on these 2 sets of tasks.

To build the set of tasks unaffected by OSS, we select all the skill-level tasks from Libero-100, the most comprehensive and diverse task suite in Libero, spanning 12 manipulation scenes and covering a wide range of object interactions and motor skills. Multi-skill tasks (e.g., "open the top drawer of the cabinet and put the bowl in it") are excluded, while single-skill tasks (e.g., "open the bottom drawer of the cabinet") are retained. Thus, the final set comprises 44 single-skill tasks.

To build the set of counterparts, we propose the **Rule-based Automatic Modification Generator (RAMG)**, which can scalably generate modified tasks. RAMG is an algorithm designed to enhance the visual diversity of skills in predefined environments through structured, rule-based modifications. Operating on PDDL files, RAMG iteratively

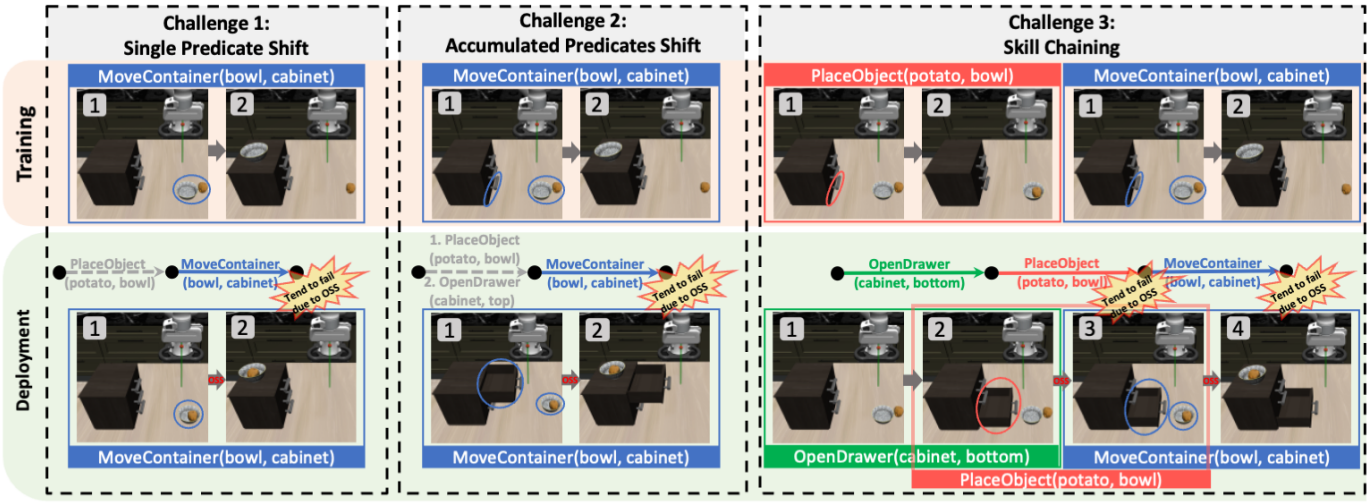

Fig. 2. This figure illustrates the three challenges of BOSS, each examining a distinct aspect of OSS, using concrete examples: `OpenDrawer(cabinet, bottom)` (green), `PlaceObject(potato, bowl)` (red), and `MoveContainer(bowl, cabinet)` (blue). Challenge 1, Single Predicate Shift (BOSS-C1), shows a case where OSS occurs due to the modification of a single predicate (i.e., the circle in the figure) caused by the effect of the previous skill (e.g., `IN(potato, bowl)`). Challenge 2, Accumulated Predicate Shift (BOSS-C2), highlights the scenario where OSS arises from multiple predicate changes (i.e., circles in the figure) due to accumulated effects from preceding skills (e.g., `IN(potato, bowl)` and `DrawerOpen(cabinet, top)`). Challenge 3, Real Long-Horizon Task (BOSS-C3), showcases how OSS impacts a real long-horizon task with three skills, where "Single Predicate Shift" and "Accumulated Predicate Shift" occur in the second skill and the third skill respectively, significantly degrading the final task performance.

modifies predicates, Ψ, by altering object positions, introducing new objects, or changing object states within the environment. It supports three types of modifications: (1) repositioning existing objects or adding external objects to specific regions, (2) changing the state of fixtures (e.g., toggling a predicate like `DrawerOpen(bottom_drawer, cabinet)`), and (3) modifying containment relations by altering predicates like `In(potato, bowl)`. These modifications adhere to dynamic constraints (e.g., newly added objects do not interfere with existing `Pre` and `Eff` conditions) and maintain logical consistency (e.g., preventing contradictions where a predicate ψ_i and its negation $\neg\psi_i$ hold simultaneously). Using deterministic yet flexible rules, RAMG provides a scalable method for generating task variations. With a single modification per task, RAMG can produce up to 1,727 modified tasks from the 44 selected tasks, and adding multiple modifications significantly increases this number. The extensive set of generated modified tasks offers flexibility and a robust foundation for designing the three challenges.

C. Challenge 1: Single Predicate Shift (BOSS-C1)

As outlined in §III, OSS arises when skill-irrelevant predicates, Ψ, are altered in ways that do not affect the feasibility of the current skill but disrupt the execution of its visuomotor policy. For example, as shown in the left box of Figure 2, the predicate `In(potato, bowl)` does not impact the feasibility of the current skill, `MoveContainer(bowl, cabinet)`, since it is absent from `Pre` and `Eff`. However, changes to this predicate can still alter the visuomotor observation space, $\mathcal{O}$, negatively affecting the skill policy's performance. In the context of skill chaining, a foundational structure in long-horizon tasks, the preceding skill is the most likely to introduce modifications in $\mathcal{O}$ that induce OSS for the subsequent skill. Therefore, we introduce a challenge, Single Predicate Shift (BOSS-C1), specifically designed to evaluate the impact of OSS caused by single-step transitions on the performance of baseline methods (see examples in Figure 2).

To investigate the impact of single-step OSS on baseline methods, we compare their performance on skills unaffected by OSS (e.g., `On(potato, table)`) versus those affected by OSS (e.g., `In(potato, bowl)`). As detailed in §IV-B, for the former, we evaluate baselines on the 44 selected tasks. For the latter, we use RAMG to generate 44 corresponding randomly modified tasks, each with a single modification applied to simulate the occurrence of OSS caused by the preceding skill.

D. Challenge 2: Accumulated Predicate Shift (BOSS-C2)

The effect of executing a skill persists until a future skill reverses it, meaning it impacts not only the immediate next skill but also multiple subsequent skills. Over the course of a skill chain, effects from numerous skills can accumulate, resulting in a more severe OSS for a future skill. Formally, given a sequence of operators $\{\texttt{op}_1, \texttt{op}_2, \ldots, \texttt{op}_t\}$, the accumulated effects at time step t are given by $\bigcup_{i=1}^{t} \texttt{Eff}_i$. As illustrated in the middle box of Figure 2, the accumulated effects, `On(plate, cabinet)` and `In(potato, bowl)`, collectively impact the current skill, `MoveContainer(bowl, cabinet)`, by modifying predicates in the observation space $\mathcal{O}$. Thus, it is valuable to study the differences between accumulated OSS and single-step OSS. To address this, we introduce another challenge, Accumulated Predicate Shift (BOSS-C2), specifically designed to analyze the impact of OSS accumulation across preceding skills (check examples in Figure 2).

Similar to BOSS-C1, we evaluate baseline performance on the next skill under two conditions: task unaffected by OSS and task modified by previous effects. Using PDDL description files, we create two sets of modified tasks, each simulating the cumulative effects of multiple preceding skills on the current skill, with one set incorporating two modifications and the other

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

three. However, multiple modifications can sometimes conflict. For example, a modification such as "place an apple inside the small bowl" could conflict with another that specifies "place a potato inside the small bowl," potentially causing the Libero environment to break due to overlapping space constraints. By iteratively invoking the proposed RAMG two or three times, we easily generate the two sets of modified tasks, leveraging RAMG's ability to ensure constraint-compliant task generation.

E. Challenge 3: Skill Chaining (BOSS-C3)

The ultimate goal of addressing the OSS problem is to improve the success of long-horizon robot tasks. To this end, we introduce a challenge, Skill Chaining (BOSS-C3), consisting of 10 long-horizon tasks, each comprising a chain of three skills (check examples in Figure 2). This challenge serves as a straightforward way to demonstrate the impact of the OSS problem on long-horizon task performance.

To construct this challenge, we manually select and combine skills from the 44 selected skill-level tasks, ensuring that OSS occurs in each skill while avoiding conflicts between modifications. We reset the robot to a neutral position after each skill to eliminate dynamic transition feasibility issues [16], ensuring the challenge focuses solely on the impact of OSS, which specifically addresses the negative effects of changes in visual observations.

V. The BOSS Experimental Results

A. Experimental Setup

1) Baselines: We select four widely used IL algorithms, representing diverse architectures and design approaches for baseline comparisons, to learn each skill and evaluate the impact of OSS on their performance in long-horizon tasks. Among these, three are Behavioral Cloning approaches from Libero, while one is a vision-language-action model.

Behavioral Cloning (BC) in Libero: We adopt three BC baselines from Libero with different visual and temporal backbones: BC-RESNET-RNN [41], BC-RESNET-T [42], and BC-VIT-T [43]. BC-RESNET-RNN uses a ResNet visual encoder and an LSTM for temporal processing. BC-RESNET-T uses the same ResNet encoder but replaces the LSTM with a transformer decoder. BC-VIT-T uses a Vision Transformer (ViT) for visual encoding and a transformer decoder for temporal processing.

OpenVLA: Applying large foundation models, such as Large Language Models (LLMs) and Vision-Language Models (VLMs), to robotics has gained popularity, leading to vision-language-action (VLA) models. Among these, OpenVLA [7], an open-sourced VLA model, outperforms other state-of-the-art methods, making it a natural choice for evaluating the OSS problem as a representative generalist policy. The language description, conditioned on each task and sourced from Libero, remains consistent, whether or not the task is affected by OSS.

2) Metrics: We mainly use two metrics in all the experiments:

Ratio Performance Delta (RPD): We define the metric "Ratio Performance Delta" as the relative change in skill success rate caused by OSS. Formally, we compute RPD $= (S_{\text{ori}} - S_{\text{mod}})/S_{\text{ori}}$, where S_{ori} is the success rate on the original skill and S_{mod} is the success rate on the modified skill affected by OSS. A positive RPD indicates that OSS negatively impacts the current skill, while an RPD less than or equal to zero suggests no negative effect. Instances of negative RPD can occur due to the inherent randomness of IL models.

Delta to Upper Bound Ratio (DUBR): Unlike BOSS-C1 and BOSS-C2, which evaluate the performance delta for individual skills, BOSS-C3 assesses the performance delta across an entire skill chain. To support this, we introduce a new metric. First, we define the "Chain Upper Bound" as the product of success rates for each skill in the chain when evaluated without OSS occurrence, representing the maximum achievable success rate in the absence of OSS. Then, we introduce the "Delta to Upper Bound Ratio", calculated as the difference between the actual success rate of completing the entire skill chain and the "Chain Upper Bound", divided by the "Chain Upper Bound", providing a normalized measure of the performance delta for the entire chain.

3) Implementation Details: For observation space design, we align with the baselines' default setups. OpenVLA uses only a third-person camera view as its observation space. To maintain consistency in visual observations, we define the observation space for BCs as a combination of third-person camera images, 7-DoF robot arm joint angles, and 2-DoF parallel gripper joint states, excluding only the wrist-camera view. For all the baselines, the action space is defined as a 7-dimensional relative Cartesian displacement (w.r.t. the gripper frame), and the control frequency is set to 20 Hz. For all the experiments, we use three random seeds and report only the averaged results across these runs.

B. Results for BOSS-C1

We evaluate baseline methods, including BC-RESNET-RNN, BC-RESNET-T, BC-VIT-T, and OpenVLA, on the 44 selected tasks and their 44 modified counterparts, calculating the Ratio Performance Delta for each baseline. The results are presented in Figure 3, where the x-axis represents the success rate on tasks unaffected by OSS, and the y-axis represents the success rate on their modified counterparts. Each point corresponds to a task pair (unaffected and modified), with darker colors indicating higher Ratio Performance Delta values. A diagonal line separates tasks: points on or above the line have non-positive Ratio Performance Delta, indicating no negative impact from OSS, while points below the line signify tasks negatively affected by OSS. As shown in Figure 3, the percentage of tasks negatively affected by OSS (points below the diagonal) is 68%, 66%, 50%, and 66% for BC-RESNET-RNN, BC-RESNET-T, BC-VIT-T, and OpenVLA, respectively. Among these tasks, the average Ratio Performance Delta is 67%, 35%, 34%, and 54% for the respective baselines, highlighting the substantial performance degradation caused by OSS. BC-VIT-T performs best, likely because its visual attention mechanism helps the policy focus on task-relevant features. In contrast, BC-RESNET-RNN performs worst, possibly because the LSTM lacks global context, making it more prone to distraction from task-irrelevant information. These results demonstrate that OSS

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

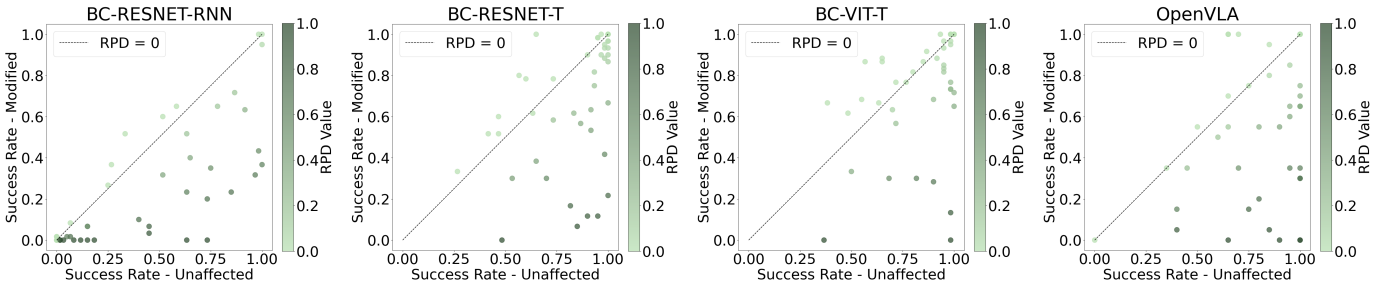

Fig. 3. This figure presents the results for BOSS-C1. In each baseline subfigure, the majority of points lie below the diagonal line, representing tasks with a positive Ratio Performance Delta, indicating that single predicate modification negatively affects tasks performance.

frequently occurs even in single-step transitions, emphasizing its potential to jeopardize the success of long-horizon tasks.

C. Results for BOSS-C2

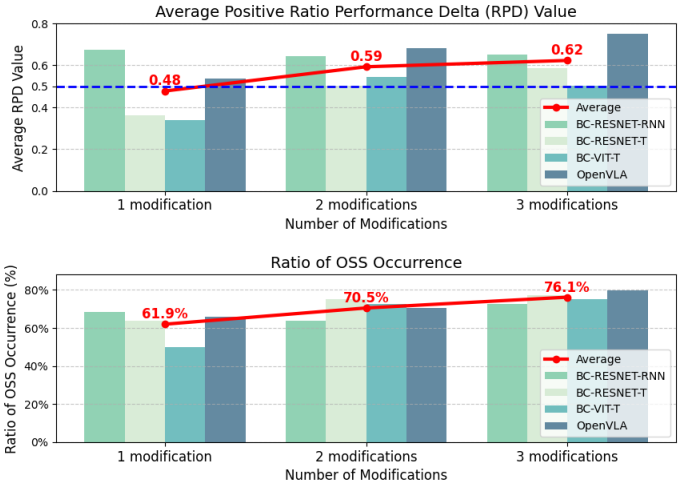

Fig. 4. This figure shows the results for BOSS-C2. Two bar charts summarize: (top) the average positive Ratio Performance Delta for sets with different numbers of modifications, and (bottom) the average ratio of OSS occurrence across sets with varying numbers of modifications. The upward trend of the red lines in both bar charts indicates the accumulation of OSS exacerbates its negative impact on long-horizon task completion in magnitude and frequency.

As detailed in §IV-D, we use the proposed RAMG to generate two sets of modified tasks: one with two modifications and another with three, simulating scenarios of accumulated modifications. Similar to BOSS-C1, we evaluate baselines on the 44 unaffected tasks and their corresponding modified counterparts. Figure 4 presents the evaluation results for BOSS-C2 alongside BOSS-C1 (i.e., single modification), yielding three sets of results for comparison. Two bar charts summarize key findings: (top) the average positive Ratio Performance Delta across the three sets and (bottom) the average occurrence ratio of OSS (i.e., cases where Ratio Performance Delta is positive). Each bar chart consists of three grouped bars, where each group corresponds to sets with one, two, or three modifications, and within each group, bars represent different baselines. Additionally, a red line denotes the average performance across all baselines for each set. In both charts, this red line follows an increasing trend, indicating that as the number of modifications grows, the negative impact on task performance intensifies, both in magnitude and frequency. Notably, across all baselines, the average Ratio Performance Delta for sets with multiple modifications exceeds 50% for all baselines (blue line in the top bar chart), signifying a substantial performance decline that is likely to cause failures in long-horizon tasks. Given that skill effect accumulation is inherent in long-horizon tasks, with modifications compounding along the skill chain, these results highlight that OSS presents a significant obstacle to achieving reliable performance.

D. Results for BOSS-C3

We include BOSS-C3 to evaluate the impact of OSS on skill chaining. As described in §IV-E, we test baselines on 10 manually designed tasks, each comprising a skill chain of three skills, and use the "Delta to Upper Bound Ratio" metric to quantify OSS's effect on performance. Figure 5 presents the results, the "Delta to Upper Bound Ratio" is shown as bar charts, where different baselines are shown in varied colors. The "Delta to Upper Bound Ratio" values are positive and high in most cases. Note that some bars for BC-RESNET-RNN show 0% values because the "Chain Upper Bound" for those tasks is already 0%, indicating that BC-RESNET-RNN fails even without OSS. These results further emphasize OSS's detrimental effect on long-horizon task completion.

VI. Can We Mitigate OSS?

Since OSS arises from visual changes introduced by preceding skills, a natural question is how to make policies more robust to these changes. In this section, we explore three straightforward strategies: (1) using robotics-specific frozen vision encoders (§VI-A), (2) switching to pointcloud-based input modalities (§VI-B), and (3) increasing data diversity through large-scale data augmentation (§VI-C).

A. Using Robotics-Specific Frozen Vision Encoders

There are two potential issues with the vision encoders used in the baselines above. First, they are not tailored for robotics tasks and may fail to learn representations that are robust to task-irrelevant changes. Second, all baselines are trained or fine-tuned end-to-end, which may lead to overfitting and performance drops during deployment. To address these issues, we evaluate two frozen, robotics-specific vision encoders: R3M [21] and LIV [22]. This results in two new variants, BC-R3M-T and BC-LIV-T, both retaining the transformer decoder for temporal processing.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

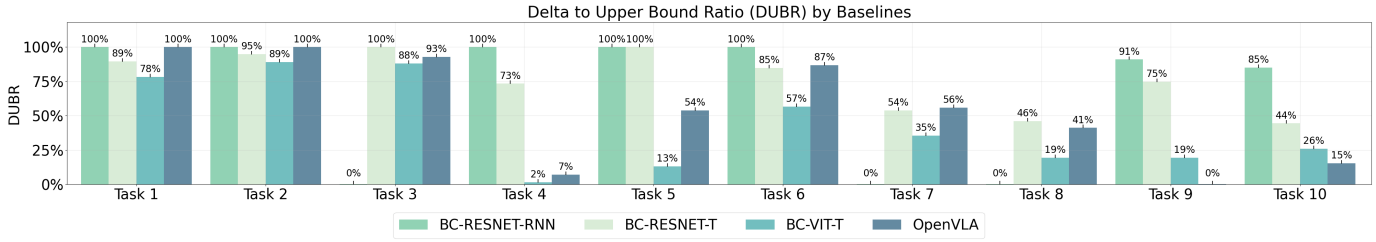

Fig. 5. This figure presents the results for BOSS-C3, where the "Delta to Upper Bound Ratio" (bars) is positive and notably high in most cases, highlighting the substantial negative impact of OSS on long-horizon task completion.

We evaluate these variants on BOSS-CH1 using the same setup as in §V-B. The percentage of skills negatively affected by OSS is 55% for BC-R3M-T and 61% for BC-LIV-T. Among these tasks, the average Ratio Performance Delta is 29% and 26% for BC-R3M-T and BC-LIV-T, respectively. These results show that frozen robotics-specific encoders fail to eliminate the impact of OSS and still suffer from broad and substantial performance drops.

B. Using 3D Pointcloud-Based Policies

Another idea is to switch to pointcloud-based policies, which have shown better generalization to viewpoint changes compared to image-based methods [44], [45]. We evaluate 3D Diffuser Actor [23], a recent state-of-the-art pointcloud policy, on BOSS-CH1. Despite its advantages, we find that 73% of the skills are still negatively affected by OSS, with an average Ratio Performance Delta of 37%. These results suggest that even policies that do not rely on images remain vulnerable to observation space shifts caused by preceding skills.

C. Using Large-Scale Data Augmentation

A third approach is to expand the training dataset by adding more diverse visual examples. The intuition is that exposing policies to a wider range of observation shifts during training may improve robustness at test time. To simulate this, we use RAMG to generate 1,727 modified versions of the 44 selected skills, each introducing a single modification. We replay demonstrations in these modified environments, filtering out failures to generate a new dataset of 57,000 demonstrations, which is nearly 30 times the size of the original Libero dataset for the 44 selected tasks (2,000 demonstrations).

TABLE I
COMPARISON OF BASELINE PERFORMANCE (SUCCESS RATE) ON SETUP A AND SETUP B. SETUP A: BASELINES TRAINED ON ORIGINAL DEMONSTRATIONS AND EVALUATED ON SKILLS AFFECTED BY OSS. SETUP B: BASELINES TRAINED ON AUGMENTED DEMONSTRATIONS AND EVALUATED ON THE SAME SKILLS AFFECTED BY OSS. THE COMPARISON REFLECTS THE DIFFERENCE IN RESULTS BETWEEN THE TWO SETUPS.

	Setup A	Setup B	Comparison (A - B)
BC-RESNET-RNN	0.25	0.31	-0.06
BC-RESNET-T	0.67	0.54	0.13
BC-VIT-T	0.74	0.64	0.10
OpenVLA	0.54	0.54	0.00

We compare two setups to evaluate whether this data augmentation helps:

- **Setup A**: Baselines are trained on the original demonstrations and evaluated on the same 44 modified skills used in BOSS-CH1.
- **Setup B**: Baselines are trained on the augmented dataset and evaluated on the same 44 modified skills.

Table I reports the averaged success rates for both setups. While BC-RESNET-RNN shows slight improvement in Setup B, the other baselines, including BC-RESNET-T, BC-VIT-T, and OpenVLA, perform similarly or worse. This suggests that data augmentation alone provides limited benefit. One possible reason is that larger datasets increase diversity but also make it harder for some baselines to generalize. Moreover, the combinatorial explosion of scene and task variations makes it impractical to cover all relevant cases through data augmentation alone, even in simulation.

VII. CONCLUSION

This work introduces and investigates Observation Space Shift (OSS), a critical issue that hinders the completion of long-horizon robot tasks. Through three challenges in BOSS, we evaluate the impact of OSS on the performance of several popular baseline algorithms and present key findings:

- **OSS is a common and severe problem**: Results on BOSS demonstrate that OSS can substantially degrade task completion, especially in long-horizon tasks where visual modifications accumulate over time.
- **Straightforward solutions may not be sufficient**: We evaluate three intuitive strategies, but none effectively resolve the OSS:
  - *Frozen robotics-specific vision encoders (R3M and LIV)*: These fail to improve robustness and still suffer from substantial performance drops.
  - *3D pointcloud-based policies (3D Diffuser Actor)*: Despite their ability to generalize to viewpoint changes, these policies remain highly sensitive to OSS.
  - *Data augmentation with diverse visual modifications*: While increasing dataset size and diversity generally improves performance, it does not fully mitigate OSS and can even degrade performance in some cases. This is likely due to the difficulty of covering the vast combinatorial space of scene variations that arise from different skill compositions in long-horizon tasks.

These findings highlight that straightforward solutions are not sufficient to mitigate the negative impact of OSS, and that current baselines lack mechanisms to handle it, emphasizing the need for tailored algorithms.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

REFERENCES

[1] Z. Liu, Q. Liu, W. Xu, L. Wang, and Z. Zhou, "Robot learning towards smart robotic manufacturing: A review," *Robotics and Computer-Integrated Manufacturing*, vol. 77, p. 102360, 2022.

[2] Z. Zaidi, D. Martin, N. Belles, V. Zakharov, A. Krishna, K. M. Lee, P. Wagstaff, S. Naik, M. Sklar, S. Choi *et al.*, "Athletic mobile manipulator system for robotic wheelchair tennis," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2245–2252, 2023.

[3] L. Chen, R. Paleja, and M. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *Conference on robot learning*. PMLR, 2021, pp. 1262–1277.

[4] Y. Yang, L. Chen, Z. Zaidi, S. van Waveren, A. Krishna, and M. Gombolay, "Enhancing safety in learning from demonstration algorithms via control barrier function shielding," in *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024, pp. 820–829.

[5] Y. Yang, B. Ikeda, G. Bertasius, and D. Szafir, "Arcade: Scalable demonstration collection and generation via augmented reality for imitation learning," *arXiv preprint arXiv:2410.15994*, 2024.

[6] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.

[7] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[8] A. Bagaria and G. Konidaris, "Option discovery using deep skill chaining," in *International Conference on Learning Representations*, 2019.

[9] G. Konidaris and A. Barto, "Skill discovery in continuous reinforcement learning domains using skill chaining," *Advances in neural information processing systems*, vol. 22, 2009.

[10] Y. Lee, S.-H. Sun, S. Somasundaram, E. S. Hu, and J. J. Lim, "Composing complex skills by learning transition policies," in *International Conference on Learning Representations*, 2019.

[11] A. Clegg, W. Yu, J. Tan, C. K. Liu, and G. Turk, "Learning to dress: Synthesizing human dressing motion via deep reinforcement learning," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–10, 2018.

[12] S. Li, S. Han, Y. Zhao, B. Liang, and P. Liu, "Auxiliary reward generation with transition distance representation learning," *arXiv preprint arXiv:2402.07412*, 2024.

[13] H. Watahiki and Y. Tsuruoka, "One-shot imitation with skill chaining using a goal-conditioned policy in long-horizon control," in *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022.

[14] J.-S. Byun and A. Perrault, "Training transition policies via distribution matching for complex tasks," *arXiv preprint arXiv:2110.04357*, 2021.

[15] M. Dalal, T. Chiruvolu, D. Chaplot, and R. Salakhutdinov, "Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks," *arXiv preprint arXiv:2405.01534*, 2024.

[16] Y. Chen, C. Wang, L. Fei-Fei, and C. K. Liu, "Sequential dexterity: Chaining dexterous policies for long-horizon manipulation," *arXiv preprint arXiv:2309.00987*, 2023.

[17] Y. Lee, J. J. Lim, A. Anandkumar, and Y. Zhu, "Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization," *arXiv preprint arXiv:2111.07999*, 2021.

[18] J. Zhang, J. Zhang, K. Pertsch, Z. Liu, X. Ren, M. Chang, S.-H. Sun, and J. J. Lim, "Bootstrap your own skills: Learning to solve new tasks with large language model guidance," *arXiv preprint arXiv:2310.10021*, 2023.

[19] J. Zhang, K. Pertsch, J. Zhang, and J. J. Lim, "Sprint: Scalable policy pre-training via language instruction relabeling," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 9168–9175.

[20] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

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

[22] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman, "Liv: Language-image representations and rewards for robotic control," in *International Conference on Machine Learning*. PMLR, 2023, pp. 23 301–23 320.

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

[24] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on robot learning*. PMLR, 2020, pp. 1094–1100.

[25] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[26] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

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

[28] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations," *arXiv preprint arXiv:2107.14483*, 2021.

[29] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 3153–3160.

[30] W. Pumacay, I. Singh, J. Duan, R. Krishna, J. Thomason, and D. Fox, "The colosseum: A benchmark for evaluating generalization for robotic manipulation," *arXiv preprint arXiv:2402.08191*, 2024.

[31] O. Ahmed, F. Träuble, A. Goyal, A. Neitz, Y. Bengio, B. Schölkopf, M. Wüthrich, and S. Bauer, "Causalworld: A robotic manipulation benchmark for causal structure and transfer learning," *arXiv preprint arXiv:2010.04296*, 2020.

[32] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7327–7334, 2022.

[33] M. Heo, Y. Lee, D. Lee, and J. J. Lim, "Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation," *arXiv preprint arXiv:2305.12821*, 2023.

[34] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," *arXiv preprint arXiv:2406.02523*, 2024.

[35] S. Zhang, P. Wicke, L. K. Şenel, L. Figueredo, A. Naceri, S. Haddadin, B. Plank, and H. Schütze, "Lohoravens: A long-horizon language-conditioned benchmark for robotic tabletop manipulation," *arXiv preprint arXiv:2310.12020*, 2023.

[36] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, no. 1, pp. 265–293, 2021.

[37] H. Guo, F. Wu, Y. Qin, R. Li, K. Li, and K. Li, "Recent trends in task and motion planning for robotics: A survey," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1–36, 2023.

[38] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[39] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.

[40] Y. Du, M. Yang, P. Florence, F. Xia, A. Wahid, B. Ichter, P. Sermanet, T. Yu, P. Abbeel, J. B. Tenenbaum *et al.*, "Video language planning," *arXiv preprint arXiv:2310.10625*, 2023.

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

[42] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," in *Conference on Robot Learning*. PMLR, 2023, pp. 1199–1210.

[43] W. Kim, B. Son, and I. Kim, "Vilt: Vision-and-language transformer without convolution or region supervision," in *International conference on machine learning*. PMLR, 2021, pp. 5583–5594.

[44] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: 3d feature field transformers for multi-task robotic manipulation," *arXiv preprint arXiv:2306.17817*, 2023.

[45] 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*. PMLR, 2023, pp. 694–710.