2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Not Throwing Away My Shot: Planning Ahead With Dual Subgoals in Long-Horizon Robot Manipulation Tasks

Longrui Chen†, Yanlong Huang, and Mehmet Dogar

***Abstract*— Policy learning often encounters difficulties in long-horizon tasks. Subgoal-conditioned policies address long-horizon problems by decomposing them into manageable segments, but they usually struggle with identifying informative subgoals. To address this limitation, we propose *PDS (planning with dual subgoal)*, an architecture that learns *short-horizon* and *low-variance* subgoals in embedding space, ensuring the planning both reachable and consistent. We begin by analyzing the impact of horizon and consistency on the performance of subgoal-conditioned policies. We evaluate the performance of commonly used subgoal definitions (time-based, visual-based, and language-based) in tasks with different lengths. Subsequently, we demonstrate that our approach, which predicts and conditions on dual subgoals, improves success rates and enhances stability across diverse tasks in simulation and real-world.**

I. Introduction

Long-horizon tasks present significant challenges in robotic manipulation. Recent imitation-learning methods, which acquire manipulation capabilities directly from data, struggle with long-horizon tasks, as the enlarged exploration space and reduced dataset coverage necessitate substantially larger amounts of data [1]. Goal-conditioned imitation learning introduces the desired goal to guide the path, but it struggles with long-horizon tasks where a single goal may not provide sufficient guidance [2]. Subgoal-conditioned approaches offer a potential solution by providing stage-specific guidance instead of solely focusing on the final goal [3], but their success hinges on accurately identifying and predicting subgoals, which is particularly challenging in dynamic environments.

As illustrated by the coffee-making task in Fig. 1, the robot is required to sequentially grasp the coffee pod, insert it into the machine, and close the lid. A goal-conditioned approach leverages the final goal state to guide the policy, whereas subgoal-conditioned methods introduce intermediate guidance signals, such as time-based [4], [5] (e.g., fixed-step lookahead), visual-based [6] (e.g., key perceptual states like grasping), or language-based subgoals [7] (e.g., states derived from task decompositions by large language models).

Although subgoal-conditioned policies are widely used for long-horizon manipulation, what constitutes an effective subgoal remains unclear. We characterize subgoals by two properties: *horizon*, the temporal distance to the current state, and *variance*, the uncertainty of the subgoal, both of which critically affect policy performance in hierarchical control [8]. We evaluate existing time-, visual-, and language-based subgoal-conditioned policies in simulation and real-world tasks, and find that short-horizon and low-variance subgoals are preferred. Importantly, these two properties do not necessarily coincide: a temporally close subgoal may exhibit high variance, while a low-variance subgoal may lie too far ahead to guide fine-grained actions. Motivated by this observation, we propose *Planning with Dual Subgoals (PDS)*, which jointly incorporates a short-horizon and a low-variance subgoal into a shared embedding, yielding more consistent and improved policy performance.

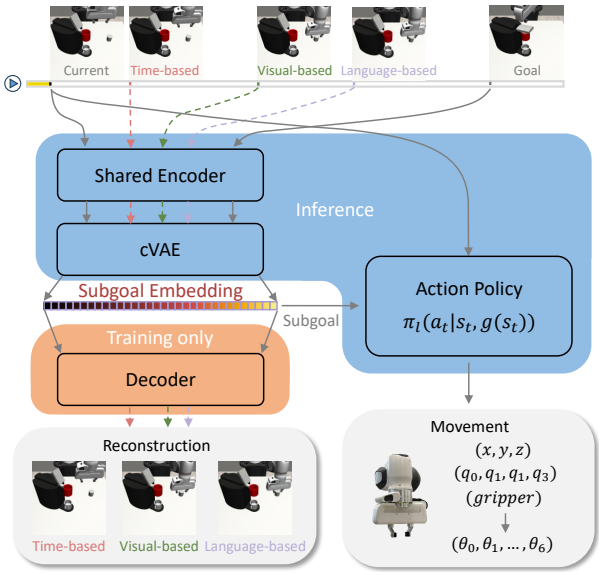

Fig. 1: Overview of the two-level subgoal-conditioned policy architecture. A conditional variational autoencoder (cVAE) is employed to learn selected subgoal's representation by predicting future subgoal from the current observation and the final goal. The resulting subgoal embedding is then provided to the action policy to guide low-level control.

The main contributions of this work are as follows:

- We analyze the influence of subgoal horizon and variance on hierarchical policy performance.
- We benchmark existing time-, visual-, and language-based subgoal policies in both simulation and real-world experiments.
- We introduce a dual-subgoal-based policy, *PDS*, which combines short-horizon and low-variance subgoals and outperforms single-subgoal policies across tasks.

Authors are with School of Computer Science, University of Leeds, Leeds, LS2 9JT, UK {sclch, scsyh, scsmrd}@leeds.ac.uk

M. Dogar was supported by the UK Engineering and Physical Sciences Research Council [EP/V052659/1]

† Corresponding author

II. Related Work

A. Subgoal-conditioned Learning

Within the learning-based domain, two main approaches have been proposed to address long-horizon tasks: skill chaining [9] and subgoal-conditioned learning [10]. Skill chaining involves identifying distinct skills from demonstrations [11] and applying these skills in appropriate contexts.

In contrast, subgoal-conditioned learning builds upon the foundations of goal-conditioned learning without distinguishing between tasks [12]. While goal-conditioned policies perform well for various tasks within similar environments when given a specific final goal, they often struggle with long-horizon tasks where a single goal may not provide adequate guidance. This limitation becomes particularly pronounced when the desired outcome diverges significantly from intermediate states [2].

To address the problem of long-horizon tasks, comprehensive subgoal-conditioned methods have been developed, such as the subgoal tree framework [13], which introduces intermediate targets to effectively guide control policies. In subgoal tree, the effectiveness of subgoal utilization primarily depends on the accuracy of the reachability measurement from one state to a desired subgoal. Several techniques have been developed to tackle this challenge [14] and can be integrated into optimization processes [15]. However, selecting appropriate subgoals for robotic manipulation remains underexplored. Common strategies focus on a single aspect of subgoal characteristics. For example, the fixed step advancement subgoal [4] uses a time-based short horizon to maintain reachability, while the uniformly sampled subgoal [5] covers certain future states. We investigate the relationship between subgoal horizon and consistency and how they impact the overall policy from different aspects.

B. Visual Representation

Effective subgoal-conditioned policies require identifying meaningful intermediate states that guide task execution. Visual representations can facilitate this by encoding rich scene and object information, enabling robots to infer subgoals in the future [16], [17]. Despite their effectiveness in task decomposition [6], raw visual embeddings often contain noise and inconsistencies that hinder reliable subgoal discovery. In this work, we leverage visual representations for subgoal identification and introduce filtering strategies to ensure that the resulting visual-based subgoals are informative and robust for policy learning.

III. Problem Definition

We leverage imitation learning to predict subgoals and capture expert behaviors. The learning problem is formulated as a Markov decision process (MDP), denoted by $M = \langle S, A, R, P \rangle$. The state space $S \subseteq \mathbb{R}^e$ includes the robot's end-effector position and orientation, alongside environmental observation. In this work, we consider, and separately evaluate, two different types of environmental observation: state-based and image-based. In the *state-based* modality, the environmental observation is object position and orientation, whereas the *image-based* modality includes two images from agent-view and robot-hand-view cameras. The action space $A \subseteq \mathbb{R}^7$ comprises incremental adjustments to the end-effector's position and rotation, as well as the gripper's state. The reward function R is designed to measure task-specific achievements. Environmental dynamics P describe state transition probabilities, where the current state $\mathbf{s}_t$ and action $\mathbf{a}_t$ determine the subsequent state $\mathbf{s}_{t+1} \sim P(\cdot \mid \mathbf{s}_t, \mathbf{a}_t)$. The expert dataset contains N task demonstrations $D = \{\tau^i\}_{i=1}^N$. Each demonstration is a sequence of state-action pairs $\tau = (\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \mathbf{a}_1, ..., \mathbf{s}_T, \mathbf{a}_T)$, where T is the length of that demonstration. The last state $\mathbf{s}_T$ is marked as the final goal, $\mathbf{g}$, of the demonstration.

We investigate how subgoal definitions influence subgoal-conditioned imitation policies. We present different subgoal definitions in Sec. IV-A. Given a subgoal definition c, we create an augmented dataset D_c, in which each demonstration is augmented such that $\tau_c = (\mathbf{s}_0, \mathbf{a}_0, \mathbf{g}_c(\mathbf{s}_0), \mathbf{s}_1, \mathbf{a}_1, \mathbf{g}_c(\mathbf{s}_1), ...)$, where $\mathbf{g}_c(\mathbf{s}_t)$ denotes subgoal for state $\mathbf{s}_t$. Each subgoal $\mathbf{g}_c(\mathbf{s}_t)$ is selected from among the future states in the same demonstration, using the subgoal definition c.

We then use the augmented dataset to train a hierarchical, subgoal-conditioned policy decomposed into two components: a *high-level subgoal planner* $\pi_h(\mathbf{g}(\mathbf{s}) \mid \mathbf{s}, \mathbf{g})$ which, given a state and final goal, predicts a subgoal $\mathbf{g}(\mathbf{s})$ for the state, and a *low-level action policy* $\pi_l(\mathbf{a} \mid \mathbf{s}, \mathbf{g}(\mathbf{s}))$ which, given a state and a subgoal, predicts an action. Our objective is to assess how the definition of subgoal c influences the overall policy performance and generalization across different tasks. Since the action policy is guided by the predicted future subgoal at each step, it does not require explicit verification of subgoal achievement.

Below, we discuss two different ways the chosen subgoal definition can affect the overall policy: subgoal horizon (Sec. III-A) and subgoal variance (Sec. III-B).

A. Subgoal Horizon

Consider two subgoals for a given state $\mathbf{s}$: a short-horizon subgoal $\mathbf{g}^{h_1}(\mathbf{s})$ and a long-horizon subgoal $\mathbf{g}^{h_2}(\mathbf{s})$, where $h_1 < h_2$ and assume both subgoals are on the optimal path to the final goal $\mathbf{g}$. We can have two policies trained with these subgoals: $\pi_l(\mathbf{a} \mid \mathbf{s}, \mathbf{g}^{h_1}(\mathbf{s}))$ and $\pi_l(\mathbf{a} \mid \mathbf{s}, \mathbf{g}^{h_2}(\mathbf{s}))$. Since there is more than one way to reach these subgoals, the trained policies will have uncertainty in predicted actions. Assuming a Gaussian form for the policies, the policy trained with the short horizon can be represented as:

$$\pi_l(\mathbf{a} \mid \mathbf{s}, \mathbf{g}^{h_1}(\mathbf{s})) \sim \mathcal{N}(\cdot, (\sigma^{h_1})^2) \tag{1}$$

where σ^{h_1} is the action standard deviation, and similarly for the policy trained with the long horizon subgoal.

Now consider the set of paths (i.e., action sequences) that connect $\mathbf{s}$ to $\mathbf{g}^{h_1}(\mathbf{s})$ and the set of paths that connect $\mathbf{s}$ to $\mathbf{g}^{h_2}(\mathbf{s})$. Since the former set is a subset of the latter, the action uncertainty of the policy trained with the shorter horizon would not be larger than the policy trained with the longer horizon:

$$\sigma^{h_1} \leq \sigma^{h_2} \tag{2}$$

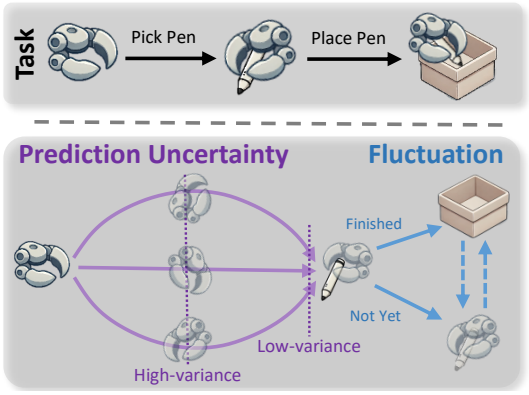

Fig. 2: Subgoal consistency problems. (Purple) Subgoal uncertainty at different variance levels. (Blue) Subgoal prediction fluctuations between the next two subgoals when the current state is near a potential subgoal (picking the pen).

Hence, in general, the action uncertainty σ^h would increase with the subgoal horizon h. As the horizon extends, the accumulated error in action prediction grows, leading to greater deviation. So, it is more effective to generate a closer subgoal rather than a distant one, thereby maximizing the expected reward $E(R)$ of the action policy.

B. Subgoal Variance

We now discuss two effects of subgoal variance at different stages: *subgoal variance* in the dataset, which reflects the variability of subgoals collected from demonstrations, and *subgoal prediction fluctuation*, which arises during inference and impacts the temporal consistency of predicted subgoals.

Subgoal Variance While we argued in the previous section that short-horizon subgoals are better for action prediction, they can be problematic in terms of subgoal prediction. Datasets commonly exhibit uncertainties in both actions and subgoals, implying that for a given state $\mathbf{s}_t$, the corresponding actions and subsequent subgoals may differ. The subgoal variance can be formally expressed as:

$$\sigma_{g_c} = E\left[(\mathbf{g}_c(\mathbf{s}) - E(\mathbf{g}_c(\mathbf{s}))(\mathbf{g}_c(\mathbf{s}) - E(\mathbf{g}_c(\mathbf{s})))^\top\right] \quad (3)$$

For example, as in Fig. 2 (Purple), using a middle-point subgoal can have high variance, while farther-away subgoals (e.g., the key state where the pen is grasped) can have low variance (i.e., better spatial consistency). Training a high-level subgoal predictor π_h under high subgoal variance poses significant challenges. When the same input state corresponds to diverse subgoal outputs, the learning process becomes unstable and difficult to converge.

Subgoal Prediction Fluctuation In addition to causing training difficulties, high subgoal variance can lead to temporal inconsistency in predictions. Even if a generative model outputs a clear subgoal at one timestep, it may produce a substantially different subgoal at the next for high subgoal variance. This *prediction fluctuation* can manifest as oscillations along the trajectory for short-horizon subgoals, or abrupt switches between distant subgoals for long horizons.

The issue of subgoal fluctuation during inference is often overlooked in existing research. This phenomenon occurs when the predicted subgoal oscillates between nearby candidates, particularly when the current state is close to a potential subgoal. The manifestation of fluctuation varies with the choice of subgoal horizon: for short horizons, the predicted subgoal tends to oscillate along the trajectory leading to oscillating behavior; for long horizons, the prediction may alternate abruptly between the current and the next subgoal which may be far away from each other (illustrated in blue in Figure 2).

We define subgoal fluctuation as the occurrence of abrupt changes in the predicted subgoal's temporal index. In fluctuation calculation, for each state, we predict the corresponding subgoal and find the closest state to each subgoal in the demonstration/testing sample. Using the state indices of each subgoal, we assess whether the predictions oscillate—i.e., increase and then decrease, or vice versa—between consecutive time steps. The fluctuation count is defined as:

$$F = \sum_{t=1}^{T-1} \mathbf{1}_A(t) \quad (4)$$

where the indicator function $\mathbf{1}_A(t)$ is defined as:

$$\begin{cases} 1, & \text{if } (I(\tilde{\mathbf{g}}(\mathbf{s}_{t-1})) < I(\tilde{\mathbf{g}}(\mathbf{s}_t)) \text{ and } I(\tilde{\mathbf{g}}(\mathbf{s}_t)) > I(\tilde{\mathbf{g}}(\mathbf{s}_{t+1}))) \\ & \quad \text{or } (\tilde{\mathbf{g}}(\mathbf{s}_{t-1})) > I(\tilde{\mathbf{g}}(\mathbf{s}_t)) \text{ and } I(\tilde{\mathbf{g}}(\mathbf{s}_t)) < I(\tilde{\mathbf{g}}(\mathbf{s}_{t+1}))) \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where $\tilde{\mathbf{g}}(\mathbf{s}_t)$ is the subgoal predicted by π_h in timestep t, $I(\tilde{\mathbf{g}}(\mathbf{s}_t))$ gives the corresponding state index (inside the demonstration trajectory) of $\tilde{\mathbf{g}}(\mathbf{s}_t)$.

C. Combining short-horizon and low-variance subgoals

We aim to obtain subgoals that are both short-horizon and low-variance. Horizon and variance, however, are two distinct properties of subgoals and do not necessarily coincide. In practice, a subgoal that is temporally close may still exhibit high variance depending on task progression, whereas a stable subgoal may lie too far in the future to effectively guide fine-grained behavior. These considerations motivate the use of dual subgoals instead of relying on a single one. Formally, given a subgoal-conditioned policy with horizon h, the ideal optimal subgoal $\bar{\mathbf{g}}^h$, and the low-level action policy:

$$\pi_l(\mathbf{a}|\mathbf{s}, \tilde{\mathbf{g}}^h), \text{where } \tilde{\mathbf{g}}^h \sim \mathcal{N}(\bar{\mathbf{g}}^h, (\sigma^h)^2) \quad (6)$$

We aim to define subgoals that maintain a short horizon, exhibit low variance, and minimize fluctuation. While using either a short-horizon subgoal or a low-variance subgoal addresses only one aspect of this goal, we propose predicting both a short-horizon subgoal $\mathbf{g}^{sh}$ and a low-variance subgoal $\mathbf{g}^{lv}$ by the high-level planner, and feeding them into the action policy, as shown below:

$$\pi_l\left(\mathbf{a} \mid \mathbf{s}, \mathbf{g}^{lv}, \mathbf{g}^{sh}\right) \pi_h\left(\mathbf{g}^{lv}, \mathbf{g}^{sh} \mid \mathbf{s}, \mathbf{g}\right) \quad (7)$$

Empirically, subgoal prediction fluctuation often correlates with different stages of the task, as reflected in the dataset

distribution. The proposed dual subgoal framework takes advantage of this property in inference. By leveraging both the current observation and the reliability of the predicted subgoals, the action policy can dynamically balance between the two subgoals—selecting the subgoal most beneficial for the current task phase—thus enhancing robustness and continuity in decision-making.

IV. Method

A. Subgoal Definitions

As discussed in Sec. III, we use different subgoal definitions to generate different augmented datasets D_c. Subgoal definitions that are commonly used can be categorized into three: time-based, visual-based, and language-based.

Time-based Subgoals We adopt two methods to define time-based subgoals: a fixed step forward from the current timestep [4] (time-I) and an equal division of the task into segments of consistent horizon length [5] (time-II). Time-based methods can be used to create short-horizon and fixed-horizon subgoals.

In $D_{time\text{-}I}$, the subgoal is defined as:

$$\mathbf{g}(\mathbf{s}_t) = \mathbf{s}_{t+n} \tag{8}$$

where n is a predetermined step interval, which is set to 10 in our experiments.

In $D_{time\text{-}II}$, the subgoal is formulated as:

$$\mathbf{g}(\mathbf{s}_t) = \mathbf{s}_m, \quad m = \min\{m \in M \mid m > t\}, \tag{9}$$

where M represents the set of subgoal time-points, which divides the task's temporal domain into equal intervals, m denotes the first time-point after t.

Visual-based Subgoals Temporally defined subgoals, though useful, often lack task-specific details and can be noisy due to trajectory variability. In contrast, visual subgoals better capture meaningful changes in task progress. To this end, we use UVD [6], referred to as Visual-I, for visual subgoal identification. UVD selects key states based on turning points in image embeddings, producing lower-variance subgoals compared to time-I.

However, despite that visual-I detects subgoals based on critical points in trajectory, it does not guarantee consistency between demonstrations and resistance to noise. To address this, we enhance visual-I by incorporating a filtering approach and generating visual-II. To filter and refine the set of subgoals, we apply K-means clustering to the subgoals identified by the visual-I across all demonstrations in the same task. The number of clusters K is set to the average number of subgoals per demonstration. This clustering groups similar subgoal candidates together, with each cluster center μ_k representing a canonical subgoal.

For each demonstration, we then identify subgoal states as those closest to the cluster centers. The resulting set of filtered subgoals G_{sub} in the dataset $D_{\text{visual-II}}$ is defined as:

$$\begin{aligned} G_{\text{sub}} = \Big\{ &\arg\min_{\mathbf{s}} \|\mathbf{s} - \mu_k\| \text{ for } k \in \{1,2,\ldots,K\} \Big\} \\ &\text{s.t. } \forall \mathbf{g}_i, \mathbf{g}_j \in G_{\text{sub}},\ \|\mathbf{g}_i - \mathbf{g}_j\| \geq \delta \end{aligned} \tag{10}$$

where μ_k denotes the center of the k-th cluster. The distance constraint δ ensures that selected subgoals are sufficiently distinct from one another temporally, preventing redundant or overly similar subgoal selections.

Language-based Subgoals The use of large language models (LLMs) in robotics has enabled the adoption of language-based subgoals, which align with how humans naturally break tasks into sequential steps when planning the future [7]. While LLMs are trained on extensive human data, the effectiveness of language-based subgoals in robotic tasks is still unclear. To investigate this, we introduce a third subgoal category, language-I.

Instead of generating subgoals directly from videos, we use text prompts via ChatGPT to identify task-relevant subgoals, followed by human annotation to label the dataset $D_{language\text{-}I}$. For example, in the threading task, the prompt "How to insert a needle on the table into a hole?" yields necessary subgoals like "1. identify the needle's position, 2. locate the hole ..." Compared to time-based and visual-based approaches, language-based methods produce fewer subgoals, as LLMs focus on essential steps.

Dual Subgoal To capitalize on the benefits of both short-horizon and low-variance subgoals, we combine dataset time-I, representing short-horizon subgoals, with other datasets for low-variance guidance, resulting in the generation of $D_{PDS\text{-}t2}$, $D_{PDS\text{-}v1}$, $D_{PDS\text{-}v2}$, and $D_{PDS\text{-}l1}$. Time-I subgoal nearly capturing all state variance in the dataset becomes the most inconsistent subgoal, which will be qualitatively evaluated in Sec. V. The augmented dataset includes state-action-(dual subgoal) sequence $\tau_c^i = (\mathbf{s}_0^i, \mathbf{a}_0^i, \mathbf{g}^1(\mathbf{s}_0^i), \mathbf{g}^2(\mathbf{s}_0^i), \ldots, \mathbf{s}_{T_i}^i, \mathbf{a}_{T_i}^i, \mathbf{g}^1(\mathbf{s}_{T_i}^i), \mathbf{g}^2(\mathbf{s}_{T_i}^i))$.

B. Policy Details

The overall architecture is shown in Fig. 1, with a conditional variational autoencoder (cVAE) being the subgoal planner and RNN as the action policy. The labeled subgoals and expert action are used to train the subgoal planner and action policy jointly.

In cVAE, the current observation s_t and goal g are used as conditioning inputs to reconstruct subgoal in training and generate subgoal embedding $\mathbf{l}_t$ in inference. To handle multimodal data distributions, we use Gaussian mixture model (GMM) prior $p_\theta(\mathbf{l}_t \mid \mathbf{s}_t, \mathbf{g})$ for cVAE, the parameters of GMM are learned. The training loss of subgoal planner is formulated as:

$$\begin{aligned} \mathscr{L}_{\pi_h} = &\mathbb{E}_{q_\phi(\mathbf{l}_t \mid \mathbf{g}(\mathbf{s}_t), \mathbf{s}_t, \mathbf{g})} \left[\log p_\theta(\mathbf{g}(\mathbf{s}_t) \mid \mathbf{l}_t, \mathbf{s}_t, \mathbf{g})\right] \\ &- \beta D_{KL}(q_\phi(\mathbf{l}_t \mid \mathbf{g}(\mathbf{s}_t), \mathbf{s}_t, \mathbf{g}) \parallel p_\theta(\mathbf{l}_t \mid \mathbf{s}_t, \mathbf{g})). \end{aligned} \tag{11}$$

Here $q_\phi(\mathbf{l}_t \mid \mathbf{g}(\mathbf{s}_t), \mathbf{s}_t, \mathbf{g})$ is the posterior distribution and $p_\theta(\mathbf{l}_t \mid \mathbf{s}_t, \mathbf{g})$ represents the likelihood. We use 64-dimensional subgoal embedding in all methods.

The timing of KL loss introduction in cVAE training is critical: applying it too early degrades reconstruction, while delaying it can lead to disorganized embeddings and overfitting. We adopt a cyclical annealing schedule [18] to balance reconstruction and regularization. To further enhance

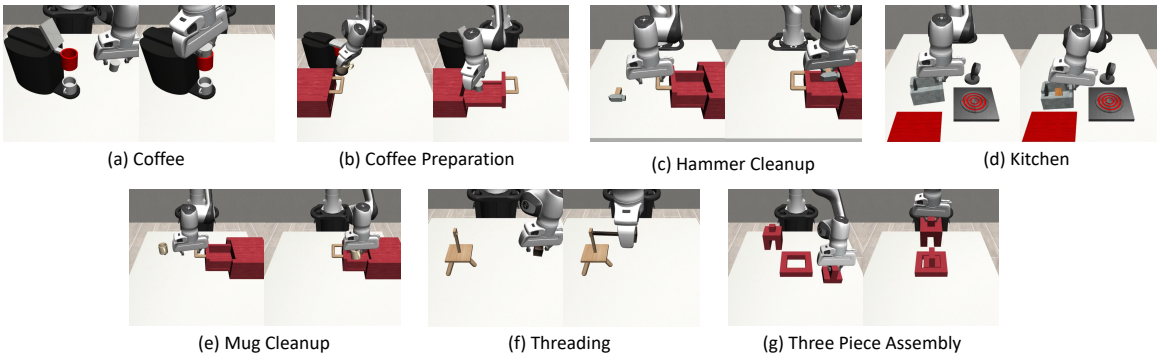

Fig. 3: Environments used to test subgoal-conditioned policies. The environments include both short-horizon and long-horizon tasks, requiring multiple skills to finish the tasks, such as pick, place, insertion, open-and-close lid, and mobile manipulation.

image realism, a GAN is incorporated, and VIP loss [17] is applied in the time-I method to maintain a smooth, goal-directed subgoal embedding space.

In the subgoal planner, a shared encoder processes observation, subgoal, and goal images, reducing model complexity and accelerating training. We evaluated R3M [16], MVP [19], and VIP [17] as pre-trained feature extractors. R3M tends to blur motion sequences, MVP can miss small object details, whereas VIP preserves finer details and is thus preferred.

We implement an action policy using an RNN combined with a GMM, which generates actions based on observation, subgoal embedding and historical data. The RNN captures temporal dependencies in the observation sequences, while GMM models expert action selection by representing the action distribution as a mixture of Gaussian components, allowing for flexible and multi-modal decision-making.

V. Results

We first evaluate the subgoal variance, fluctuation, and performance of subgoal-conditioned strategies in simulation. Then we conduct a real-world experiment to test policy performance. Finally, we use the Berkeley UR5 demonstration dataset [20] to qualitatively analyze the simulation and real-world difference of subgoal prediction. Overall, our results show that, using dual subgoals that are short-horizon, low-variance, and low-fluctuation improves the policy performance. Please also see attached video for examples of predicted subgoals.

A. Simulation Environment

To evaluate the subgoal-conditioned policy, we utilize the MimicGen environment [21] as our testing platform. Built upon RoboMimic [22] and RoboSuite [23], MimicGen supports both short-horizon and long-horizon tasks, as illustrated in Fig. 3. We take the 'Core' expert dataset from the official source, containing 1000 demonstrations/trajectories for each task. It is crucial to note that the dataset from MimicGen is suboptimal (machine-generated) and includes failure cases, which present significant challenges for subgoal detection.

Selected robotic tasks include long-horizon activities like coffee preparation and kitchen tasks, each comprising approximately 15 subgoals identified by multi-task visual decomposer (UVD) [6], and several short-horizon tasks. The coffee preparation, threading, and coffee tasks demand precise manipulation for insertion. The three-piece assembly task involves two similar blocks, while the kitchen task features red fires on the stove when in use, making it difficult for visual subgoal generation. We present an example subgoal prediction of simulation in Fig. 4 (a).

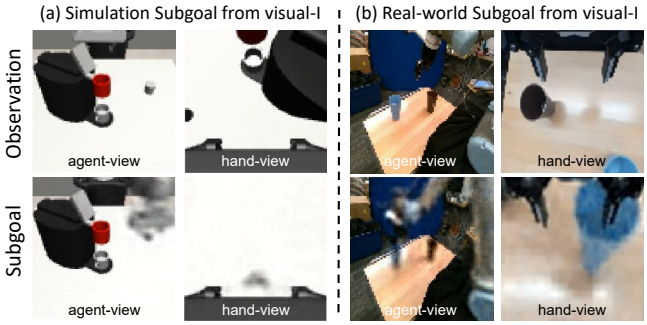

Fig. 4: The figure shows subgoal predictions using Visual-I. The top row displays the current state observation, while the bottom row shows the subgoal predicted by the model. Part (a) on the left corresponds to simulation, and part (b) on the right corresponds to data from the Berkeley UR5 dataset.

B. Subgoal Variance and Fluctuation

A similarity threshold is applied to group similar states from different trajectories. We compute the average subgoal variance (Eq. 3) of similar states for all subgoal definitions by measuring pixel-level differences (with values ranging from 0 to 255). The results show that time-I has the highest variance at 1540, time-II has 447, visual-I has 413, visual-II has 363, and language-I has the lowest variance at 280.

Time-I's high variance indicates significant uncertainty, while time-II, with a variance of 447, is more stable but still higher than the visual-based methods. Language-I, with the lowest variance of 280, shows the highest consistency across all tasks.

After the training of subgoal predictor, we measured subgoal fluctuation of all subgoal-conditioned policies. For dual-subgoal conditioned policies, the fluctuation was counted

whenever either of the two subgoals fluctuated. The average fluctuation count of single-subgoal conditioned policies is 49% higher than dual-subgoal conditioned policies. The dual subgoal strategy improves robustness by preventing the model from overfitting to the fluctuations of a single subgoal, thus enhancing its overall stability in dynamic environments.

C. Subgoal Fluctuation with a Real Large Dataset

We trained real-world subgoal predictors π_h using an extended version of the Berkeley UR5 dataset, selecting four tasks with more than 200 demonstrations each. Unlike simulation environments, real-world data captures fine-grained visual details—such as textures and background clutter—which introduce noise and increase the complexity of visual reconstruction. We present an example subgoal prediction from this dataset in Fig. 4 (b).

To assess the temporal consistency of subgoal prediction, we analyzed the fluctuation of predicted subgoals across time. Due to the lower frame rate in real-world data collection, task durations were notably shorter than in simulation. On average, subgoal fluctuation was reduced by 10% compared to methods conditioned on a single static subgoal. Remarkably, despite increased randomness in real-world executions, the fluctuation was 90% lower than in simulated environments. This indicates that while complex textures introduce reconstruction challenges, they may also promote the learning of more stable subgoal representations.

D. Single-task Evaluation in Simulation

In the single-task evaluations, we evaluated ten different policies (1 Goal-Conditioned, rest subgoal-conditioned). Each policy was tested with a total of 500 rollouts using five different seeds. To analyze variability, the 500 rollouts were divided into five groups by seed, and the success rate was computed for each group to obtain the standard deviation. We use the Goal-Conditioned policy $\pi_l(\mathbf{a} \mid \mathbf{s}, \mathbf{g})$ as our baseline, which is implemented as a GMM-RNN action policy, with the final state utilized as the goal. The results are shown in Table I. For each task, the highest success rates are highlighted in bold, while the lowest success rates are marked in red. Only values that are significantly higher or lower than the others (exceeding one standard deviation) are highlighted.

Overall (averaged across all tasks, bottom row), dual-subgoal policies perform better. The baseline Goal-Conditioned policy ranks the lowest in both the state-based and image-based spaces. All subgoal-conditioned policies, regardless of subgoal horizon and consistency, outperform the goal-conditioned policy. This improvement is due to the shorter planning paths in the low-level action policies.

Time-I performs well by maintaining a close subgoal in both state-based and image-based spaces. However, in the image-based space, time-I struggles to capture detailed scene information in environments such as fire in the kitchen and object difference in three-piece assembly due to large subgoal variance.

Compared to other long-horizon subgoals, the primary challenge with time-II lies in subgoal consistency. Despite having the highest number of subgoals, time-II exhibits the lowest success rate and the most frequent poor performance cases in the image-based space among subgoal-conditioned policies. Task demonstrations show high variability in trajectory length and motion duration, causing time-II's subgoals to lack consistency across demonstrations. This inconsistency leads to challenges for the subgoal planner, resulting in fluctuations and blurry reconstructions.

In contrast to time-II, visual-I benefits from selecting subgoals at key states rather than at fixed intervals, leading to better performance in the image-based space, although with some variance in the number and length of subgoals. The filtering mechanism implemented in visual-II stabilizes performance compared to visual-I (which has more worst-case scenarios) and improves performance in the state-based space. Language-I, with the largest subgoal horizon but most consistent subgoal, achieves the same performance as visual-I. These comparisons demonstrate that consistent subgoals can improve policy performance by aiding the subgoal planner in generating stable outputs.

By learning temporal features of short-horizon subgoals and spatial features of low-variance subgoals, the dual subgoal embedding space enables the PDS method to achieve the most stable performance. With the same latent size, network, and hyperparameters, the dual subgoal setting improves the performance of both visual-based and language-based policies across all dimensions. Despite sparse subgoals, the combination with Time-I in PDS-l1 yields the highest score and the most consistent performance in both state-based and image-based spaces.

We observe an overall drop in success rate from time-I to PDS-t2, indicating that simply combining subgoals does not guarantee improvement. The key lies in combining short-horizon and low-variance subgoals. PDS-t2 performs exceptionally well on the three-piece assembly task, despite time-I and time-II performing poorly individually. This result highlights the complementary nature of the two subgoals: time-I's short-horizon focus struggles with long-term dependencies, while time-II lacks fine-grained temporal information. The combination of both subgoals provides a more comprehensive understanding of the task.

Regarding the combination of dual low-variance subgoals (such as visual-I with visual-II), we validated there is no improvement made by dual subgoals. Combining two visual subgoals lead to redundancy rather than improvement.

In summary, subgoal horizon is a critical factor in subgoal-conditioned policies. Short-horizon subgoals work well for simple tasks but underperform in complex tasks where the subgoal planner cannot capture all details. Training a subgoal planner with consistent subgoals helps maintain performance across tasks. Learning features of short-horizon and low-variance subgoals ensures good and consistent performance, particularly in image-based space.

E. Multi-task Evaluation in Simulation

The challenge of learning a multi-task policy lies in its requirement to effectively handle both the subgoal planner

TABLE I: Success rate and standard deviation of baseline and different subgoal-conditioned policies.

Environment	Type	Goal-Cond.	Time-I [4]	Time-II [5]	Visual-I [6]	Visual-II	Language-I	PDS-t2	PDS-v1	PDS-v2	PDS-l1
Coffee	state-based	0.92±0.02	0.93±0.03	0.90±0.02	0.91±0.03	0.95±0.02	0.93±0.02	0.90±0.03	**0.97±0.01**	**0.95±0.02**	0.93±0.04
	image-based	0.84±0.02	0.88±0.04	0.87±0.03	**0.96±0.02**	0.93±0.02	0.90±0.03	0.81±0.04	0.86±0.05	0.91±0.05	**0.97±0.01**
Coffee Preparation	state-based	0.42±0.03	0.43±0.06	0.39±0.03	0.22±0.02	**0.46±0.08**	0.37±0.03	0.39±0.03	0.37±0.04	**0.46±0.08**	**0.49±0.07**
	image-based	0.37±0.06	0.81±0.04	0.63±0.04	0.63±0.03	0.68±0.04	0.76±0.06	0.66±0.03	**0.86±0.02**	0.85±0.04	**0.87±0.04**
Hammer Cleanup	state-based	0.95±0.02	1.00±0.00	1.00±0.01	1.00±0.00	1.00±0.00	1.00±0.01	0.99±0.02	0.99±0.01	1.00±0.00	0.99±0.01
	image-based	**0.99±0.01**	**1.00±0.01**	0.94±0.02	0.97±0.02	0.89±0.03	**1.00±0.01**	0.99±0.01	**1.00±0.02**	0.97±0.01	**0.97±0.03**
Kitchen	state-based	0.97±0.01	1.00±0.01	0.99±0.01	1.00±0.00	1.00±0.00	0.99±0.01	1.00±0.01	1.00±0.00	1.00±0.00	1.00±0.00
	image-based	0.80±0.06	0.71±0.06	**0.99±0.01**	0.84±0.03	0.97±0.01	**0.99±0.01**	0.97±0.02	0.98±0.01	0.96±0.02	**1.00±0.00**
Mug Cleanup	state-based	0.30±0.02	0.63±0.04	0.47±0.05	0.58±0.05	0.64±0.04	0.41±0.03	0.47±0.05	**0.70±0.06**	**0.69±0.04**	0.63±0.03
	image-based	0.44±0.03	**0.51±0.03**	0.16±0.02	0.34±0.04	0.41±0.03	0.34±0.03	0.48±0.06	0.43±0.03	0.48±0.03	0.40±0.02
Threading	state-based	0.60±0.06	0.71±0.05	0.74±0.05	0.65±0.05	0.70±0.02	0.66±0.03	0.74±0.05	0.67±0.05	0.68±0.03	**0.86±0.04**
	image-based	0.68±0.02	0.83±0.05	0.72±0.04	0.77±0.03	0.69±0.05	0.62±0.05	0.48±0.08	0.65±0.02	0.82±0.03	**0.89±0.03**
Three Piece Assembly	state-based	0.33±0.04	0.55±0.04	**0.57±0.05**	0.53±0.07	**0.59±0.04**	0.54±0.04	0.50±0.05	0.53±0.07	0.49±0.03	0.54±0.06
	image-based	0.25±0.03	0.26±0.02	0.24±0.01	**0.38±0.02**	0.32±0.03	0.29±0.03	**0.39±0.04**	0.37±0.06	0.31±0.05	0.34±0.04
AVERAGE SUCCESS RATE	state-based	0.64±0.08	**0.75±0.05**	0.72±0.05	0.70±0.07	0.74±0.04	0.70±0.07	0.71±0.06	**0.75±0.06**	**0.75±0.05**	**0.78±0.05**
	image-based	0.62±0.07	0.72±0.06	0.65±0.10	0.70±0.06	0.70±0.06	0.70±0.08	0.68±0.05	0.74±0.06	**0.76±0.06**	**0.78±0.08**

Note: Citations for Time-I, Time-II, and Visual-I indicate the source of the subgoal definition, not the policy framework.

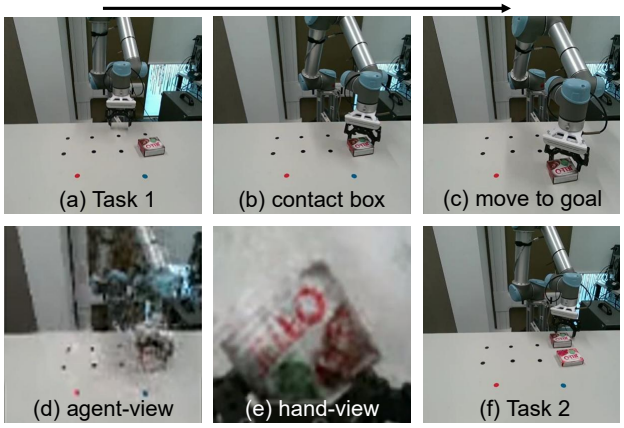

Fig. 5: Real-world non-prehensile pushing tasks. (a) Task 1: push the box to the blue point. (b, c) Execution of Task 1, where the robot first establishes contact with the box (b) and then pushes it to the goal location (c). (d, e) Reconstructed subgoal predictions for Task 1 corresponding to the initial state in (a). (f) Task 2: pushing the box to the blue target while avoiding a second box (the obstacle).

and the action policy, especially when there is significant variability between tasks. The multi-task policy is trained using the same network architecture and parameters as those used for single-task policies. We restrict the evaluation to the image-based space due to the variable observation dimensions in state-based spaces, which depend on the number of objects involved in each task. To ensure training convergence, the number of training episodes for the multi-task policy is set to be ten times that of the single-task policy, and the embedding space is expanded by the same factor. Each policy is evaluated in each environment over 300 rounds.

In the multi-task setting, the overall success rate reduces compared to the single-task setting. The goal-conditioned policy, serving as our baseline, achieved an average success rate of 0.23(±0.03) across all tasks. Among single-subgoal

TABLE II: Success rates of different policies in two real-world non-prehensile pushing tasks.

Policy	Task 1	Task 2
Goal-Cond.	1/20	0/20
Time-I	3/20	0/20
Time-II	3/20	1/20
Visual-I	3/20	2/20
Visual-II	5/20	3/20
Language-I	8/20	4/20
PDS-t2	6/20	4/20
PDS-v1	7/20	4/20
PDS-v2	8/20	6/20
PDS-l1	**13/20**	**8/20**

policies, Time-I demonstrated the highest performance, with an average success rate of 0.29(±0.03). Other single-subgoal policies exhibit similar performance to the baseline. Time-II and Visual-I showed slightly higher success rate variance due to subgoal variance. Within the dual-subgoal category, PDS-v1 achieved the best performance, reaching the highest success rate of 0.38 (±0.03), followed by PDS-l1 at 0.30 (±0.02). Other dual-subgoal policies attained success rates 0.28 (±0.02/0.03). The results suggest that using dual subgoals enhances policy performance, particularly in multi-task settings. The underlying principle of using short-horizon subgoals appears to hold in both single-task and multi-task scenarios. However, in multi-task training, exposure to diverse subgoals may occasionally promote better generalization, even when subgoal definitions vary across tasks.

F. Real-world Experiments

We evaluated different subgoal definitions in two real-world non-prehensile pushing scenarios (Fig. 5). Task 1 involved short-horizon planning, while Task 2 required long-horizon planning with obstacle avoidance. In both settings, the robot started away from a jello box, made initial contact, and pushed it to a designated blue goal point. Compared to simulation, real-world experiments exhibit greater ran-

domness, particularly in the initial object positions, which are randomized to evaluate policy generalization. In more complex environments, the improvement achieved by dual-subgoal policies becomes more pronounced.

Table II summarizes the results. Each policy was trained on 20 demonstrations. Each of the ten policies were then tested for the two tasks for 20 trials, resulting in a total of 400 real-robot policy executions. In Task 1, all PDS variants outperformed the baseline, with PDS-l1 achieving the highest rate (13/20). Dual-subgoal conditioning notably improved performance compared to single-modality approaches. In Task 2, performance decreased for all policies due to the increased planning horizon and obstacle complexity. Nevertheless, PDS-l1 maintained the top success rate (8/20), indicating better generalization to complex real-world scenarios.

An example of subgoal visualization for visual-I, taken at state Fig. 5 (a), is shown in Fig. 5 (d,e). The predicted subgoal in the initial state correctly shows how the robot should contact with the box to push to the goal position. The edges of the robot and the object exhibit blurring around their target positions due to dataset variance. This visual artifact can be mitigated with a larger and more diverse training dataset and further analysis is conducted in the following section from the Berkeley UR5 subgoal prediction analysis.

Two key challenges emerged in real-world subgoal-conditioned policy execution: (1) object displacement due to unintended sliding during grasping, and (2) focal degradation when the object or camera is positioned too closely. To address these issues, we propose: (1) incorporating a sliding-aware subgoal refinement mechanism informed by gripper state feedback, and (2) deploying an auto-tuned focal adjustment system to dynamically mitigate focal loss during close-range manipulation.

VI. Conclusion

This work evaluated the impact of subgoal horizon and subgoal consistency in subgoal-conditioned policies. Short-horizon subgoals reduce the decision-making horizon for the action policy, while low-variance subgoals are more predictable and stable. We introduced a new subgoal-conditioned policy, Plan with Dual Subgoal, which integrates the advantages of both short-horizon and low-variance subgoals. With small adjustments the system architecture, the policy demonstrates improved and stable performance.

Acknowledgments

The illustrative image of the robot hand, pen, and box in Fig. 2 was generated using ChatGPT. The AI-generated content is used solely for visualization purposes and does not affect any experimental data, results, or conclusions reported in this paper.

References

[1] P. Englert, A. Paraschos, M. P. Deisenroth, and J. Peters, "Probabilistic model-based imitation learning," *Adaptive Behavior*, vol. 21, no. 5, pp. 388–403, 2013.

[2] E. Chane-Sane, C. Schmid, and I. Laptev, "Goal-conditioned reinforcement learning with imagined subgoals," in *International conference on machine learning*. PMLR, 2021, pp. 1430–1440.

[3] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

[4] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei, "Learning to generalize across long-horizon tasks from human demonstrations," in *Robotics: Science and Systems (RSS)*, 2020.

[5] P.-C. Ko, J. Mao, Y. Du, S.-H. Sun, and J. B. Tenenbaum, "Learning to act from actionless videos through dense correspondences," in *The Twelfth International Conference on Learning Representations*, 2024.

[6] Z. Zhang, Y. Li, O. Bastani, A. Gupta, D. Jayaraman, Y. J. Ma, and L. Weihs, "Universal visual decomposer: Long-horizon manipulation made easy," *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6973–6980, 2023.

[7] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg, "Text2motion: From natural language instructions to feasible plans," *Autonomous Robots*, vol. 47, no. 8, pp. 1345–1365, 2023.

[8] B. Aceituno and A. Rodriguez, "A hierarchical framework for long horizon planning of object-contact trajectories," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 189–196.

[9] U. A. Mishra, S. Xue, Y. Chen, and D. Xu, "Generative skill chaining: Long-horizon skill planning with diffusion models," in *Proceedings of The 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Tan, M. Toussaint, and K. Darvish, Eds., vol. 229. PMLR, 06–09 Nov 2023, pp. 2905–2925.

[10] Z. Huang, Y. Lin, F. Yang, and D. Berenson, "Subgoal diffuser: Coarse-to-fine subgoal generation to guide model predictive control for robot manipulation," *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16 489–16 495, 2024.

[11] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," in *International Conference on Learning Representations (ICLR)*, 2019.

[12] S. Cheng and D. Xu, "League: Guided skill learning and abstraction for long-horizon manipulation," 2023.

[13] T. Jurgenson, O. Avner, E. Groshev, and A. Tamar, "Sub-goal trees a framework for goal-based reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 5020–5030.

[14] S. Nair and C. Finn, "Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation," in *International Conference on Learning Representations (ICLR)*, 2020.

[15] B. Brito, M. Everett, J. P. How, and J. Alonso-Mora, "Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4616–4623, 2021.

[16] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *Conference on Robot Learning*, 2022.

[17] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang, "Vip: Towards universal visual reward and representation via value-implicit pre-training," in *International Conference on Learning Representations (ICLR)*, 2023.

[18] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, "Cyclical annealing schedule: A simple approach to mitigating kl vanishing," in *North American Chapter of the Association for Computational Linguistics*, 2019.

[19] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik, "Masked visual pre-training for motor control," 2022.

[20] L. Y. Chen, S. Adebola, and K. Goldberg, "Berkeley UR5 demonstration dataset," https://sites.google.com/view/berkeley-ur5/home.

[21] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. S. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," in *Conference on Robot Learning*, 2023.

[22] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Mart'in-Mart'in, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning*, 2021.

[23] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv preprint arXiv:2009.12293*, 2020.