Video Glue: Strengthening the Interface between Video Prediction and Low-Level Control for Hierarchical Imitation Learning

Kyle B. Hatch[1] **Ashwin Balakrishna**[1] **Oier Mees**[2] **Suraj Nair**[1] **Seohong Park**[2] **Blake Wulfe**[1] **Masha Itkina**[1] **Benjamin Eysenbach**[3] **Sergey Levine**[2] **Thomas Kollar**[1] **Benjamin Burchfiel**[1]

[1]Toyota Research Institute [2]UC Berkeley [3]Princeton University

Abstract: While internet-scale training of video predictive models has the potential to provide robot learning systems with one type of generalization (to the millions of scenarios seen in the training data), there are other facets of generalization that are not addressed directly by internet-scale models. For example, they may generate photorealistic yet infeasible frames and they assume that the low-level controller can generalize from in-domain images to the generated images. This paper takes aim at these latter two facets of generalization, providing an interface to effectively "glue together" language-conditioned video prediction models with low level goal-conditioned policies. We present Video Glue, a simple way to achieve this by (1) filtering out subgoals that do not lead to task progress and (2) improving the robustness of goal-conditioned policies to subgoals with hallucinated artifacts. Experiments show that Video Glue achieves a new state-of-the-art on the CALVIN simulation benchmark and significantly outperforms other generalist robot policies both in simulation experiments and across 5 language-conditioned manipulation tasks in physical experiments.

Keywords: Hierarchical Imitation Learning, Video Prediction

1 Introduction

The success of internet-scale foundation models in computer vision and NLP is exciting for the robot learning community because it takes aim at one of the key challenges in our field: generalization. However, despite major strides in collecting increasingly large action-labeled robot manipulation datasets [1, 2, 3], video datasets (without actions) from the internet are vastly larger, and have the potential to provide generalization and common sense capabilities to robotic control policies. Building an effective robot learning system requires the sorts of generalization provided by internet-scale data sources, but also requires robustly generating low-level robot commands.

This paper studies the problem of hierarchical imitation learning, using a framework pioneered by prior work [4, 5]. Modern hierarchical imitation learning algorithms typically leverage a video predictive model trained on internet-scale data to predict subgoal images, and a low level policy which translates these subgoal images into a sequence of motor commands [6, 7]. While this general approach has seen significant success in prior work on robotic manipulation [8, 6, 9, 7, 10, 11], the interface between the high-level planner generating subgoals and the low-level policy that must reach these subgoals can be brittle. State-of-the-art (SOTA) video prediction models are relatively good at generating likely subgoal images given a language prompt describing the task, but these subgoal generations may not necessarily be functionally useful for control. First, these subgoals may not actually make progress towards completing the task, even if they are likely to occur at some point when the task is performed. Second, even if the generated subgoals lead to task progress, they may contain hallucinated artifacts which make them unsuitable for control.

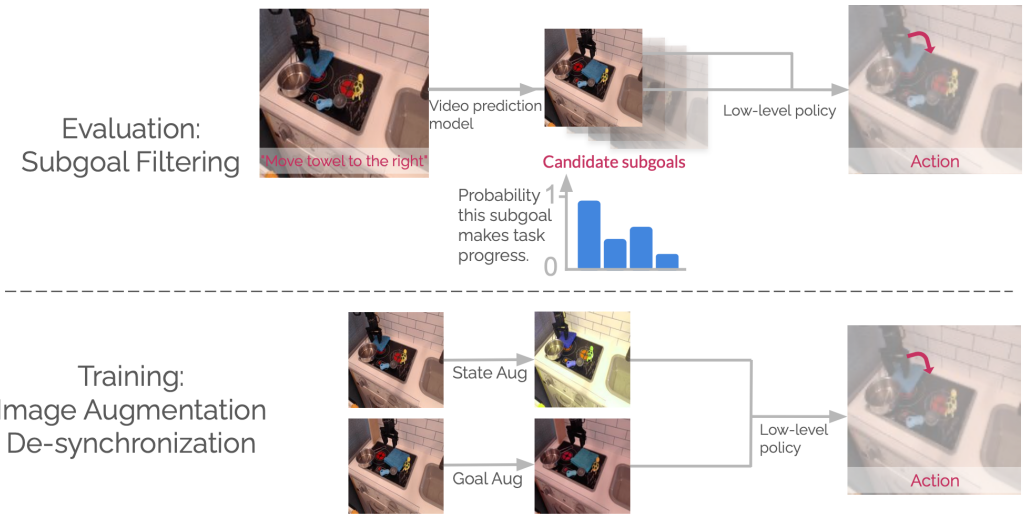


Figure 1: Video Glue Video Glue has two components: subgoal filtering (top) and augmentation de-synchronization (bottom). We consider a language-conditioned video prediction model that can generate multiple subgoals based on the current image observation. We train a subgoal filtering classifier that identifies which subgoal is most likely to make progress towards completing the desired language instruction. We then select the most likely subgoal under our filter and pass it to the low-level policy along with the current state to generate an action for the robot to execute. Since the current image observation and the generated subgoal fed to the low-level policy are sampled in different ways, we explicitly de-synchronize (bottom) the image-augmentations applied during training to the current state (State Aug) and the sampled goal (Goal Aug). This serves to robustify both the low-level policy and subgoal filter to artifacts in generated candidate subgoals.

We propose Video Glue (Figure 1), a method to *robustly* "Glue" together video prediction models to a low-level robotic control policy. Our method is based on two components. **First**, we filter out generated subgoals that are physically inconsistent with the commanded language instruction. We sample a number of candidate subgoals from our video prediction model and select the subgoal with the highest likelihood of making progress towards the desired task. We do this by training a classifier on robot data inspired by [12], which predicts the likelihood of the transition between the current state and a given subgoal resulting in progress towards completing the desired task. Then, this classifier can be used to select the subgoal with the highest likelihood, ensuring that the low-level policy plans towards subgoals likely to make meaningful task progress. **Second**, we present a new data augmentation method to robustify the low-level policy and our classifier to hallucinated artifacts in the predicted subgoals. Experiments on the CALVIN [13] simulation benchmark and 5 language-conditioned tasks on the Bridge V2 physical robot platform [14] suggest that Video Glue improves upon prior SOTA methods across the board while adding minimal additional algorithmic complexity. Notably, applying Video Glue to SuSIE [6], a prior SOTA method for hierarchical imitation learning, yields a new SOTA on the CALVIN simulation benchmark.

2 Related Work

Generative Models for Robotic Control: Prior works have explored diverse ways to leverage generative models, such as diffusion models [15, 16] and Transformers [17], for robotic control. They have employed highly expressive generative models, potentially pre-trained on Internet-scale data, for low-level control [18, 19, 20, 21], data augmentation [22, 23, 24], object detection [25, 26], semantic planning [27, 28, 29, 30, 31], and visual planning [8, 6, 9, 7, 10, 11]. Among them, our work is most related to prior works that employ image or video prediction models to generate intermediate subgoal images for the given language task [8, 6, 9, 7, 10, 11]. Similarly to ours, these works use diffusion models to convert language instructions into visual subgoal plans, which are then fed into low-level subgoal-conditioned policies to produce actions. While sensible, these previous works directly use potentially hallucinated subgoals without any validation, which often

leads to low-level policy failures, as shown in our experiments (Section 5). In this work, we propose techniques to filter out infeasible subgoals and enhance the robustness of low-level goal-reaching, and demonstrate that these techniques improve performance substantially.

Rejection Sampling: One of our key ideas in this paper is based on rejection sampling, where we sample multiple subgoal proposals from a video prediction model and pick the best one based on a learned critic function. The idea of test-time rejection sampling has been widely used in diverse areas of machine learning, such as filtering-based action selection in offline reinforcement learning (RL) [32, 33, 34, 35], response verification in natural language processing [36, 37, 38], and planning and exploration in robotics [39, 29, 30, 40, 41]. Closely related to ours, previous works in robotics have proposed several ways to filter out infeasible plans generated by pre-trained foundation models [39, 29, 30, 40, 42]. Unlike these works, we focus on filtering visual subgoals instead of language plans [29, 40, 42], and do not involve any planning procedures [30] or structural knowledge [39].

Goal-Conditioned Policy Learning: Our method is broadly related to goal-conditioned RL [43, 44, 45], language-conditioned policy learning [46, 47, 48, 49, 50], and hierarchical control [4, 5]. Most prior works in hierarchical policy learning either train a high-level policy from scratch that produces subgoals or latent skills [51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65] or employ subgoal planning [66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79]. Unlike these works, we do not train a high-level subgoal prediction model from scratch nor involve a potentially complex planning procedure. Instead, we sample multiple potential subgoals from a pre-trained (or potentially fine-tuned) video prediction model and simply pick the best one based on a critic function. Among hierarchical RL methods, perhaps the closest work to ours is IRIS [80], which trains a conditional variational autoencoder to generate subgoal proposals and selects the best subgoal that maximizes the task value function. While conceptually similar, our method is different from IRIS in two ways: we do not assume a task reward function, and we focus on improving the robustness of low-level policies to hallucinated visual subgoals via critic-based subgoal filtering and data augmentation.

3 Preliminaries

We consider the same problem setting as [6], where the goal is for a robot to perform a task described by some previously unseen language command l. To do this, we consider the same three dataset categories as in [6]: (1) language-labeled video clips $\mathcal{D}_l$ which contain no robot actions; (2) language-labeled robot data $\mathcal{D}_{l,a}$ that includes both language labels and robot actions; (3) unlabeled robot data that only includes actions $\mathcal{D}_a$. The dataset $\mathcal{D}_{l,a}$ consists of a set of trajectory and task language pairs, $\{(\tau^n, l^n)\}_{n=1}^N$, and a trajectory contains a sequence of state, $s_t^n \in \mathcal{S}$, and action, $a_t^n \in \mathcal{A}$, pairs, $\tau^n = (s_0^n, a_0^n, s_1^n, a_1^n, \ldots)$. Given these datasets, we assume access to two learned modules: (1) a subgoal generation module from which we can sample multiple possible future subgoals for our robot to reach and (2) a low-level goal-reaching policy which can plan to the generated subgoals. Our contribution is a set of approaches to robustify the interface between these two modules in order to enable more performant hierarchical planning.

Our work makes no assumptions on the structure of the subgoal generation module or low-level goal-reaching policy, and there are multiple reasonable choices for subgoal generation, but we study training video prediction models as in [7] and image-editing models as in [6] on $\mathcal{D}_l \cup \mathcal{D}_{l,a}$. We define subgoals, $g \in \mathcal{G}$, as image or video samples from these models. To reach the generated subgoals, we study either training an inverse dynamics model as in [9] to densely plan between generated images, or training a goal-conditioned policy as in [6] to plan towards longer-horizon subgoals.

4 Video Glue

Video Glue employs two methods to improve the robustness of the interface between a language-conditioned video-prediction model that predicts subgoals for the robot to reach and a low-level policy that is trained to output actions to reach those subgoals. In Section 4.1, we propose a simple method to filter subgoals that do not make progress towards completing the task specified by lan-

guage instruction l. Then, in Section 4.2, we describe a data augmentation strategy that improves the robustness of the low-level policy to subgoals with hallucinated artifacts.

4.1 Subgoal Filtering

Although the video prediction models we consider are fine-tuned on robot data, a common failure mode we observe is that predicted subgoals sometimes do not correspond to states that would make progress towards completing the task specified by language instruction l. We hypothesize that this is due to the significant distribution shift between the Internet data these video prediction models are trained on and the robot data they are finetuned on. Pre-training on Internet data allows them to produce subgoals with highly realistic appearances, but the limited amount of robot data makes the likelihood of hallucinating plausible subgoals for different tasks relatively high.

To address this challenge, we take inspiration from [12] and train a classifier $f_\theta(s, g, l)$ on $\mathcal{D}_{l,a}$ that predicts the probability that the transition between the current image observation s and the next subgoal g makes progress towards completing language instruction l. We use the same training objective as [12], sampling positive examples of state-goal transitions for l from the set of trajectories that successfully complete the instruction. We sample negatives in the following three ways:

1. **Wrong Instruction:** (s, g, l') where l' is sampled from a different transition than s and g.
2. **Wrong Goal Image:** (s, g', l) where g' is sampled from a different transition than s and l.
3. **Reverse Direction**: (g, s, l), where the order of the current image observation and the subgoal image have been switched. This is important for learning whether a candidate goal image is making temporal progress towards completing the language instruction.

Unlike [12], which always includes the first state s_0 in all sampled state-goal pairs, we sample state-goal pairs throughout the trajectory to make the learned classifier more amenable for hierarchical planning (see Appendix A for additional training details). Given a set of K subgoals predicted by the video-prediction model, Video Glue selects the subgoal with the highest progress probability and passes that subgoal to the low-level policy for conditioning.

4.2 Image Augmentation De-synchronization

While the method proposed in Section 4.1 can increase robustness to predicted subgoals which do not make task progress, predicted subgoals with hallucinated artifacts can also cause significant challenges for both the low-level control policy and the subgoal filter we propose. We find that a simple data augmentation trick significantly increases the robustness of both models to such subgoals.

Applying image augmentation procedures such as random cropping or color jitter during training is a common approach in computer vision [81] to improve the robustness of learned models to distribution shifts between their training and evaluation domains. More formally, let ϕ be the set of image augmentation parameters to be randomly sampled from space Φ, $p_\Phi(\cdot)$ be some probability distribution over Φ, and let $\hat{\phi} \sim p_\Phi(\cdot)$ be some realization of augmentations sampled from $p_\Phi(\cdot)$. Typically, for each training sample, a different value $\hat{\phi}$ is applied during training to train a model robust to any injected augmentation in the space Φ.

For both our low-level goal-conditioned policy and subgoal filter, each training sample includes two images: the current state s and the corresponding goal g. Standard practice is to sample augmentation parameters $\hat{\phi}$ and apply them to all images in a given training sample [6, 82], which corresponds to applying $\hat{\phi}$ to both s and g. However, when using a video prediction model for subgoal generation, at inference time the low-level policy and subgoal filter will see goals generated by the video prediction model while the current states will simply be camera observations. Thus, to encourage robustness to this distribution shift, we sample separate augmentation parameters for s and g, denoted by $\hat{\phi}_s$ and $\hat{\phi}_g$. This forces the low-level policy and subgoal filter to be robust to artifacts in the generated goals that fall within the distribution of image augmentations captured by Φ. Random cropping and color jitter comprise our space of augmentations (see Appendix B for details).

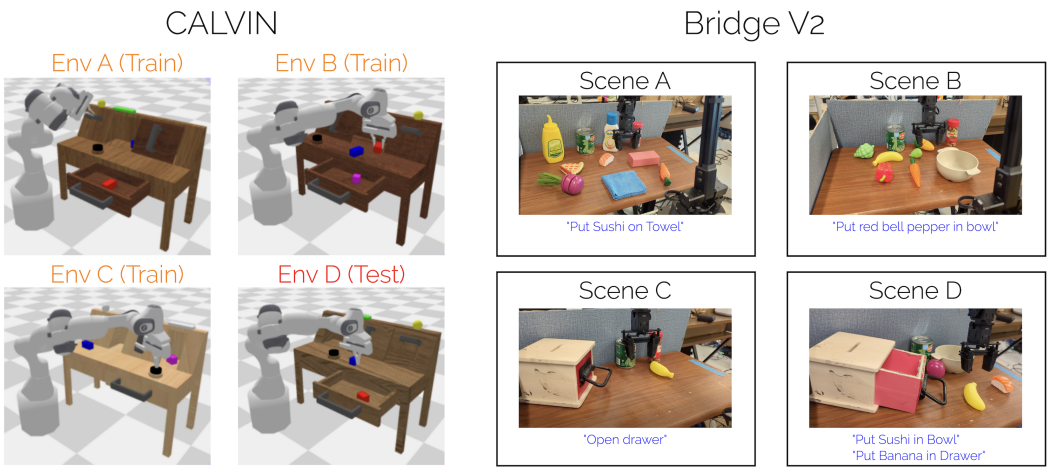


Figure 2: Experimental Domains Simulation Environments (Left): Train/test environments in the CALVIN simulation benchmark. The environments each have different table textures, furniture positionings, and initial configurations of the colored blocks. Each environment contains 34 tasks, each with an associated language instruction. Physical Environments (Right): We consider 4 test scenes in the Bridge V2 robot platform with 5 total language instructions. These test scenes contain similar objects/language instructions as those in the training data, but are generally more cluttered than the scenes in the Bridge V2 dataset, making it critical that generated subgoals focus on the right object and do not degrade in quality due to distractors in the scene.

5 Experimental Evaluation

We study the degree to which Video Glue improves existing hierarchical imitation learning algorithms across a number of tasks in simulation and physical experiments, and analyze the influence of each component of Video Glue on task performance. We also perform extensive qualitative analysis to study where Video Glue shines and probe its limitations in Appendix C.

5.1 Experimental Domains

We evaluate our method on the CALVIN [13] simulation benchmark and the Bridge V2 [14] physical experiment setup with a WidowX250 robot. We use relative cartesian position control for all tasks.

Simulation Experiment Setup: Simulation experiments are performed in the CALVIN [13] benchmark, which focuses on long-horizon language-conditioned robot manipulation. We follow the same protocol as in [6], and train on three environments (A, B, and C) and test policies on a fully unseen environment (D). Each environment contains a Franka Emika Panda robot arm that is placed in front of a desk with a variety of different objects and is associated with 34 possible tasks (Figure 2).

Physical Experiment Setup: For physical experiments, we utilize the same datasets as in [6] for training both the high-level video prediction model and the low-level goal-conditioned policy. The Bridge V2 dataset contains 45K language-annotated trajectories, which are used for the language-labeled robot dataset $\mathcal{D}_{l,a}$. The remaining 15K trajectories are used for the action-only dataset $\mathcal{D}_a$.

As in [6], we use a filtered version of the Something-Something dataset [83] with the same filtering scheme as in [6] (resulting in 75K video clips) as our video-only dataset $\mathcal{D}_l$. We test our policies on 5 tasks in 4 different cluttered table top scenes (Figure 2) on the Bridge V2 physical robot platform.

5.2 Comparison Algorithms

To evaluate Video Glue's performance, we study the impact of applying it to two SOTA hierarchical imitation learning algorithms which use vision and language pre-training: SuSIE [6] and UniPi [7]. Then, to evaluate the importance of hierarchy more generally, we also compare Video Glue to a flat language-conditioned diffusion policy (LCBC Diffusion Policy). Finally, we consider ablations where we separately study the impact of each of our proposed contributions: subgoal filtering (Sec-

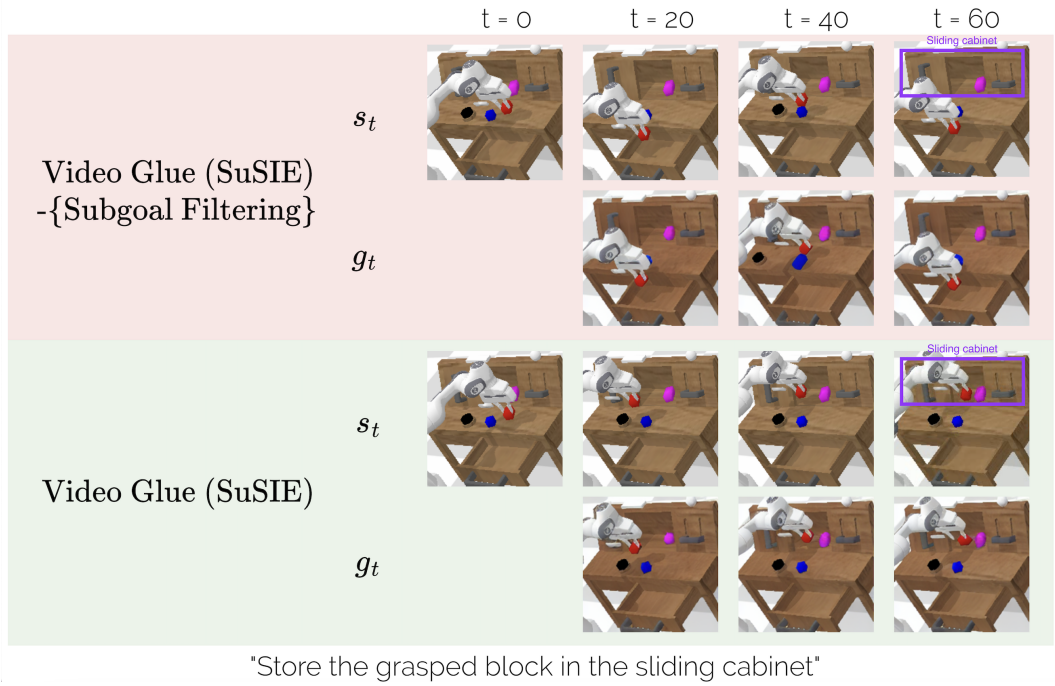


Figure 3: Video Glue Subgoal Filtering We visualize the difference in policy rollouts of Video Glue applied to SuSIE with (Video Glue (SuSIE)) and without (Video Glue (SuSIE)-{Subgoal Filtering}) subgoal filtering. In each case we visualize the states reached every 20 timesteps (top row) and the corresponding predicted subgoals (bottom row) because SuSIE predicts goal images every 20 timesteps. We find that without subgoal filtering the selected goals are often inconsistent with the commanded task, and do not make progress towards placing the grasped block in the sliding cabinet. However, when subgoal filtering is used, the goal images clearly make iterative progress towards task completion, resulting in a successful rollout.

tion 4.1) and de-synchronizing augmentations (Section 4.2). For physical experiments, we additionally consider a comparison to RT-2-X [2, 20], which is trained on the Open X-Embodiment dataset (which includes the Bridge V2 dataset). We do not compare to RT-2-X in simulation experiments because we only have API access to RT-2-X, and thus cannot finetune it on data from CALVIN for a fair comparison.

1. **LCBC Diffusion Policy:** Low-level language-conditioned behavior cloning diffusion policy [19] trained only on robot trajectories with language annotations. We use the same implementation as in [6].
2. **RT-2-X [20]:** A SOTA language-conditioned Vision-language-action model trained on the Open X-Embodiment dataset [2] (which includes the entirety of the Bridge V2 dataset)[1].
3. **SuSIE [6]:** A method which finetunes InstructPix2Pix [84], an image-editing diffusion model, to generate subgoal images given the current image observation. Low-level control is performed using a goal-conditioned policy. For SuSIE and all methods that build on it, we predict subgoals 20 steps in the future as in the original paper.
4. **UniPi [7]:** A method which finetunes a language-conditioned video prediction model on robot data and then uses an inverse dynamics model for low-level goal reaching. For UniPi and all methods that build on it, we predict video sequences of 16 frames. As the original UniPi model is not publicly available, we re-implement UniPi based on the implementation from [6]. Note that while we are unable to recreate the same UniPi performance on CALVIN as in [6], our primary focus is not on comparing UniPi with SuSIE, but rather on comparing UniPi and SuSIE with and without the addition of Video Glue.

[1]Note that RT-2-X is not goal-conditioned and is trained on a slightly older version of the Bridge V2 dataset where the first action was a zero. To partially account for this discrepancy, when querying actions from the RT-2-X API, we use the second most likely action from the API instead of the most likely action.

5. **Video Glue (SuSIE / UniPi):** Video Glue applied on top of either SuSIE or UniPi. For all experiments we implement the subgoal filtering step by sampling 8 subgoals from the high-level video prediction model and selecting amongst them. We directly filter the subgoal images generated by the SuSIE model. We filter the video sequences generated by the UniPi model based on the final frame of each sequence. See Appendix D for the ablations on the number of subgoals used for filtering.
6. **Video Glue (SuSIE / UniPi) -{Aug De-sync}:** Video Glue applied on top of either SuSIE (S) or UniPi (U) without augmentation desynchronization.
7. **Video Glue (SuSIE / UniPi) -{Subgoal Filtering}:** Video Glue applied on top of either SuSIE (S) or UniPi (U) without subgoal filtering.

5.3 Experimental Results

Method	Tasks completed in a row					
	1	2	3	4	5	Avg. Len.
LCBC Diffusion Policy	68.5%	43.0%	22.5%	11.0%	6.8%	1.52
SuSIE	89.8%	75.0%	57.5%	41.8%	29.8%	2.94
Video Glue (SuSIE) -{Subgoal Filtering}	95.2%	84.0%	69.5%	56.0%	46.2%	3.51
Video Glue (SuSIE) -{Aug De-sync}	88.5%	75.5%	56.2%	43.0%	32.5%	2.96
Video Glue (SuSIE)	**95.2%**	**88.5%**	**73.2%**	**62.5%**	**49.8%**	**3.69**

Table 1: Video Glue (SuSIE): Simulation Results Success rates on the validation tasks from the D environment of the CALVIN Challenge averaged across 4 random seeds. Video Glue (SuSIE) significantly outperforms all other methods, yielding an average task length that is the new SOTA on the CALVIN benchmark.

Method	Tasks completed in a row					
	1	2	3	4	5	Avg. Len.
UniPi	22.0%	2.5%	0.5%	0.2%	0.0%	0.25
Video Glue (UniPi) -{Subgoal Filtering}	22.0%	4.0%	0.2%	0.0%	0.0%	0.26
Video Glue (UniPi) -{Aug De-sync}	**43.8%**	**17.5%**	**4.2%**	**0.5%**	**0.0%**	**0.66**
Video Glue (UniPi)	40.2%	13.5%	5.8%	0.5%	0.0%	0.60

Table 2: Video Glue (UniPi): Simulation Results Success rates on the validation tasks from the D environment of the CALVIN Challenge averaged across 4 random seeds. Video Glue (UniPi) significantly outperforms UniPi, but subgoal filtering has a much larger influence on performance than augmentation de-synchronization.

	Task	RT-2-X	SuSIE	Video Glue (SuSIE)
Scene A	Put Sushi On Towel	4/10	6/10	**10/10**
Scene B	Put Red Bell Pepper in Bowl	0/10	5/10	**6/10**
Scene C	Open Drawer	4/10	6/10	**8/10**
Scene D	Put Sushi in Bowl	5/10	5/10	**8/10**
	Put Banana in Drawer	4/10	7/10	**9/10**

Table 3: Bridge V2 Physical Experiments Results: Success rates across 5 tasks on 4 physical robot scenes (Figure 2). We find Video Glue applied to SuSIE outperforms both RT-2-X and SuSIE across all tasks. Video Glue (SuSIE) demonstrates particularly large benefits in cluttered scenes like Scene A (Figure 2), in which there are multiple other objects that could be confused for the target object.

Simulation Experiments: We present results for Video Glue and comparisons on the CALVIN benchmark when Video Glue is applied to SuSIE (Table 4) and UniPi (Table 5). Results suggest that de-synchronizing augmentations between the current and goal observations leads to a significant improvement in performance, increasing average task length from **2.94** to **3.51** for Video Glue

applied to SuSIE (SuSIE → Video Glue (SuSIE) -{Subgoal Filtering}) and **0.25** to **0.26** for Video Glue applied to UniPi (UniPi → Video Glue (UniPi) -{Subgoal Filtering}). Interestingly, applying augmentation de-synchronization has essentially no effect when Video Glue is applied to UniPi, but a very significant effect when Video Glue is applied to SuSIE. We hypothesize that this is because UniPi uses an inverse dynamics model as the low-level policy, and thus has much more short-horizon subgoals than SuSIE (1-step subgoals). This may make the UniPi low-level policy more robust to minor, irrelevant differences between the current state and the generated subgoal.

Adding subgoal filtering further improves average task sequence length from **3.51** to **3.69** for Video Glue applied to SuSIE (Video Glue (SuSIE) -{Subgoal Filtering} → Video Glue (SuSIE)) and **0.26** to **0.60** for Video Glue applied to UniPi (Video Glue (UniPi) -{Subgoal Filtering} → Video Glue (UniPi)). See Appendix C for qualitative examples of generated subgoals for a subset of the tasks in addition to their scores under our subgoal filtering method. Interestingly, we find that for SuSIE, applying subgoal filtering without de-synchronizing augmentations yields little improvement in performance (SuSIE → Video Glue (SuSIE) -{Aug De-sync}). This suggests that unless the low-level policy is made robust to artifacts in generated subgoals, simply selecting the most task relevant subgoal is insufficient to improve task performance.

Physical Experiments: We present results (Table 3) comparing Video Glue (SuSIE) to RT-2-X and SuSIE across 5 environments on the Bridge V2 robot platform which require interacting with a number of objects on a cluttered table (Figure 2). Video Glue applied to SuSIE outperforms both RT-2-X and SuSIE across all tasks. We find that Video Glue (SuSIE) particularly shines in very cluttered scenes such as Scene A, in which there are many opportunities for generated goal images to attend to the wrong object or degrade in quality due to the added complexity of the scene. See Appendix C for qualitative examples of success and failure cases of Video Glue in physical experiments.

6 Limitations

While Video Glue increases the performance of existing hierarchical imitation learning algorithms, it still has a number of limitations. One limitation of Video Glue is its computational cost at inference time. Generating multiple subgoal images or videos from which to select limits policy execution frequency. This issue is likely to be mitigated in the future as more efficient models become available, and as hardware continues to improve. Another limitation is that while the data augmentation method we use (Section 4.2) does improve robustness to some hallucinations, it is does not have any semantic grounding. As a result, the low-level policy is still susceptible to hallucinations which introduce new objects in the scene or change relative object orientations.

7 Conclusion

We present Video Glue, a method for better aligning video prediction models and low-level control policies for hierarchical imitation learning. Our key insight is that while video prediction models can produce highly realistic subgoal images for goal-conditioned policy learning, these generated images are also prone to hallucinations and can be inconsistent with the task the robot is commanded to perform. Video Glue provides two simple ideas to address these challenges, leading to a significant increase in performance over prior work across simulation experiments on the CALVIN simulation benchmark and on 5 different tasks in physical experiments.

There are a number of exciting opportunities for future work. One avenue would be to explore training video prediction models for subgoal generation on a broader distribution of robot data, such as the data available in the Open-X embodiment dataset [2], to see if this can lead to higher quality subgoals for the low-level policy. Another interesting direction would be to filter subgoals passed to the low-level policy based on the capability of the low-level policy to actually achieve them. One possible way to instantiate this is to train a goal-conditioned value function for the low-level policy and use it to evaluate the feasibility of a commanded subgoal.

Acknowledgments

If a paper is accepted, the final camera-ready version will (and probably should) include acknowledgments. All acknowledgments go at the end of the paper, including thanks to reviewers who gave useful comments, to colleagues who contributed to the ideas, and to funding agencies and corporate sponsors that provided financial support.

References

[1] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. In *Conference on Robot Learning (CoRL)*, 2019.

[2] O. X.-E. Collaboration, A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Kolobov, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Büchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. V. Frujeri, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Yang, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. B. Amor, H. I. Christensen, H. Furuta, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Bohg, J. Bingham, J. Wu, J. Gao, J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silvério, J. Hejna, J. Booher, J. Tompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Lekkala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Mart'in-Mart'in, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Pang, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Dou, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, Z. Fu, and Z. Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. 2024.

[3] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, D. A. Herrera, M. Heo, K. Hsu, J. Hu, D. Jackson,

C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

[4] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[5] P.-L. Bacon, J. Harb, and D. Precup. The option-critic architecture. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[6] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[7] Y. Du, S. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[8] I. Kapelyukh, V. Vosylius, and E. Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.

[9] 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.

[10] A. Ajay, S. Han, Y. Du, S. Li, A. Gupta, T. Jaakkola, J. Tenenbaum, L. Kaelbling, A. Srivastava, and P. Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36, 2024.

[11] J. Gao, K. Hu, G. Xu, and H. Xu. Can pre-trained text-to-image models generate visual goals for reinforcement learning? *Advances in Neural Information Processing Systems*, 36, 2024.

[12] S. Nair, E. Mitchell, K. Chen, B. Ichter, S. Savarese, and C. Finn. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. *Conference on Robot Learning (CoRL)*, 2021.

[13] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. In *IEEE Robotics and Automation Letters (RAL)*, 2021.

[14] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, P. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

[15] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[16] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[18] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[19] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[20] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[21] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[22] Z. Mandi, H. Bharadhwaj, V. Moens, S. Song, A. Rajeswaran, and V. Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

[23] Z. Chen, S. Kiami, A. Gupta, and V. Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[24] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[25] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, S. Kirmani, B. Zitkovich, F. Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[26] A. Peng, I. Sucholutsky, B. Z. Li, T. R. Sumers, T. L. Griffiths, J. Andreas, and J. A. Shah. Learning with language-guided state abstractions. *arXiv preprint arXiv:2402.18759*, 2024.

[27] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

[28] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[29] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pages 287–318. PMLR, 2023.

[30] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *Autonomous Robots*, 47(8):1345–1365, 2023.

[31] Z. Wang, S. Cai, G. Chen, A. Liu, X. Ma, and Y. Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

[32] S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

[33] S. K. S. Ghasemipour, D. Schuurmans, and S. S. Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021.

[34] H. Chen, C. Lu, C. Ying, H. Su, and J. Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

[35] P. Hansen-Estruch, I. Kostrikov, M. Janner, J. G. Kuba, and S. Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[36] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[37] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[38] A. Hosseini, X. Yuan, N. Malkin, A. Courville, A. Sordoni, and R. Agarwal. V-star: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*, 2024.

[39] W. Liu, Y. Du, T. Hermans, S. Chernova, and C. Paxton. Structdiffusion: Language-guided creation of physically-valid structures using unseen objects. *arXiv preprint arXiv:2211.04604*, 2022.

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

[41] A. Z. Ren, J. Clark, A. Dixit, M. Itkina, A. Majumdar, and D. Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *Robotics Science and Systems (RSS)*, 2024.

[42] Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*, 2023.

[43] L. P. Kaelbling. Learning to achieve goals. In *IJCAI*, volume 2, pages 1094–8. Citeseer, 1993.

[44] T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

[45] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

[46] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:25–55, 2020.

[47] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[48] O. Mees, L. Hermann, and W. Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters (RA-L)*, 7(4):11205–11212, 2022.

[49] O. Mees, J. Borja-Diaz, and W. Burgard. Grounding language with visual affordances over unstructured data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023.

[50] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[51] J. Schmidhuber. Learning to generate sub-goals for action sequences. In *Artificial neural networks*, pages 967–972, 1991.

[52] P. Dayan and G. E. Hinton. Feudal reinforcement learning. *Advances in neural information processing systems*, 5, 1992.

[53] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

[54] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International conference on machine learning*, pages 3540–3549. PMLR, 2017.

[55] A. Levy, G. Konidaris, R. Platt, and K. Saenko. Learning multi-level hierarchies with hindsight. *arXiv preprint arXiv:1712.00948*, 2017.

[56] O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

[57] O. Nachum, S. Gu, H. Lee, and S. Levine. Near-optimal representation learning for hierarchical reinforcement learning. *arXiv preprint arXiv:1810.01257*, 2018.

[58] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

[59] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

[60] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play. In *Conference on Robot Learning (CoRL)*, pages 1113–1132. PMLR, 2020.

[61] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard. Latent plans for task agnostic offline reinforcement learning. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, Auckland, New Zealand, 2022.

[62] T. Zhang, S. Guo, T. Tan, X. Hu, and F. Chen. Generating adjacency-constrained subgoals in hierarchical reinforcement learning. *Advances in neural information processing systems*, 33: 21579–21590, 2020.

[63] K. Pertsch, Y. Lee, and J. Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pages 188–204. PMLR, 2021.

[64] E. Chane-Sane, C. Schmid, and I. Laptev. Goal-conditioned reinforcement learning with imagined subgoals. In *International Conference on Machine Learning*, pages 1430–1440. PMLR, 2021.

[65] S. Park, D. Ghosh, B. Eysenbach, and S. Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36, 2024.

[66] N. Savinov, A. Dosovitskiy, and V. Koltun. Semi-parametric topological memory for navigation. *arXiv preprint arXiv:1803.00653*, 2018.

[67] B. Eysenbach, R. R. Salakhutdinov, and S. Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

[68] S. Nair and C. Finn. Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation. *arXiv preprint arXiv:1909.05829*, 2019.

[69] S. Nasiriany, V. Pong, S. Lin, and S. Levine. Planning with goal-conditioned policies. *Advances in Neural Information Processing Systems*, 32, 2019.

[70] Z. Huang, F. Liu, and H. Su. Mapping state space using landmarks for universal goal reaching. *Advances in Neural Information Processing Systems*, 32, 2019.

[71] C. Hoang, S. Sohn, J. Choi, W. Carvalho, and H. Lee. Successor feature landmarks for long-horizon goal-conditioned reinforcement learning. *Advances in neural information processing systems*, 34:26963–26975, 2021.

[72] J. Kim, Y. Seo, and J. Shin. Landmark-guided subgoal generation in hierarchical reinforcement learning. *Advances in neural information processing systems*, 34:28336–28349, 2021.

[73] L. Zhang, G. Yang, and B. C. Stadie. World model as a graph: Learning latent landmarks for planning. In *International conference on machine learning*, pages 12611–12620. PMLR, 2021.

[74] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine. Rapid exploration for open-world navigation with latent goal models. *arXiv preprint arXiv:2104.05859*, 2021.

[75] K. Fang, P. Yin, A. Nair, and S. Levine. Planning to practice: Efficient online fine-tuning by composing goals in latent space. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4076–4083. IEEE, 2022.

[76] J. Li, C. Tang, M. Tomizuka, and W. Zhan. Hierarchical planning through goal-conditioned offline reinforcement learning. *IEEE Robotics and Automation Letters*, 7(4):10216–10223, 2022.

[77] J. Kim, Y. Seo, S. Ahn, K. Son, and J. Shin. Imitating graph-based planning with goal-conditioned policies. *arXiv preprint arXiv:2303.11166*, 2023.

[78] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard. Latent plans for task-agnostic offline reinforcement learning. In *Conference on Robot Learning*, pages 1838–1849. PMLR, 2023.

[79] K. Fang, P. Yin, A. Nair, H. R. Walke, G. Yan, and S. Levine. Generalization with lossy affordances: Leveraging broad offline data for learning visuomotor tasks. In *Conference on Robot Learning*, pages 106–117. PMLR, 2023.

[80] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

[81] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *International Conference on Intelligent Robots and Systems*, 2017.

[82] C. Zheng, B. Eysenbach, H. Walke, P. Yin, K. Fang, R. Salakhutdinov, and S. Levine. Stabilizing contrastive rl: Techniques for offline goal reaching. *arXiv preprint arXiv:2306.03346*, 2023.

[83] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, and et al. The" something something" video database for learning and evaluating visual common sense. In *IEEE international conference on computer vision (ICCV)*, 2017.

[84] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[85] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

A Classifier Training

Training objective: The classifier is trained using binary cross-entropy loss:

$$\mathcal{J}(\theta) = \mathop{\mathbb{E}}_{(s,g,l)\sim D_{l,a}}[\log(f_\theta(s,g,l))] + \mathop{\mathbb{E}}_{(s',g',l')\sim N(D_{l,a})}[1 - \log(f_\theta(s',g',l'))],$$

where $D_{l,a}$ is the language-annotated dataset that consists of trajectory and language task pairs, and N is a function for generating negative examples from the dataset. Given a dataset $D_{l,a}$, N generates negatives from $D_{l,a}$ in the following ways:

1. **Wrong Instruction:** (s, g, l') where l' is sampled from a different transition than s and g.
2. **Wrong Goal Image:** (s, g', l) where g' is sampled from a different transition than s and l.
3. **Reverse Direction**: (g, s, l), where the order of the current image observation and the subgoal image have been switched.

Across all our experiments, we sample 50% of each training batch to be positive examples and 50% of each training batch to be negative examples. Of the negative examples, 40% are "wrong instruction", 40% are "reverse direction", and 20% are "wrong goal image".

Goal sampling: In a given training tuple (s_t, g, l), g is sampled by taking the goal image from the s_{t+k}, where k is a uniformly sampled integer from 16 to 24.

Network architecture and training hyperparameters: The classifier network architecture consists of a ResNet-34 encoder from [14], followed by a two-layer MLP with layers of dimension 256. Separate encoders are used to encode the image observations and the goal images (parameters are not shared between the two). Both of these encoders use FiLM conditioning [85] after each residual block to condition on the language instruction. Classifier networks are trained using a learning rate of 3×10^{-4} and a batch size of 256 for $100,000$ gradient steps. A dropout rate of 0.1 is used.

B Image Augmentations

During training of low-level policy networks and classifier networks, we apply the following augmentations to the image observations and the goal images, in the following order:

1. Random Resized Crop:
   - scale: $(0.8, 1.0)$
   - ratio:$(0.9, 1.1)$
2. Random Brightness Shift:
   - shift ratio: 0.2
3. Random Contrast:
   - Contrast range: $(0.8, 1.2)$
4. Random Saturation:
   - Saturation range: $(0.8, 1.2)$
5. Random Hue:
   - shift ratio: 0.1

Figure 4 visualizes examples from the Bridge dataset before and after augmentations are applied:

Before Augmentation

After Augmentation

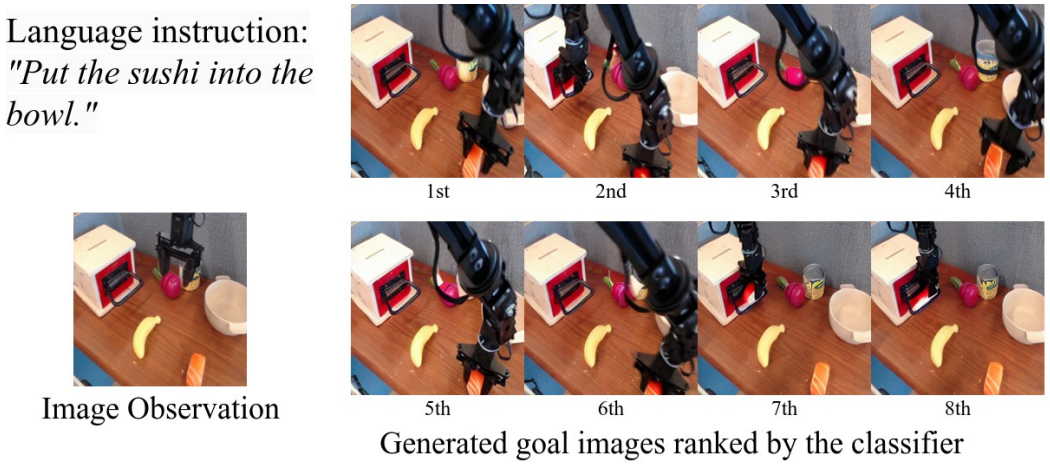
Figure 4: Image augmentation examples Examples of images from the Bridge dataset before and after having the image augmentations applied to them that are used during policy and classifier training.

C Qualitative Analysis

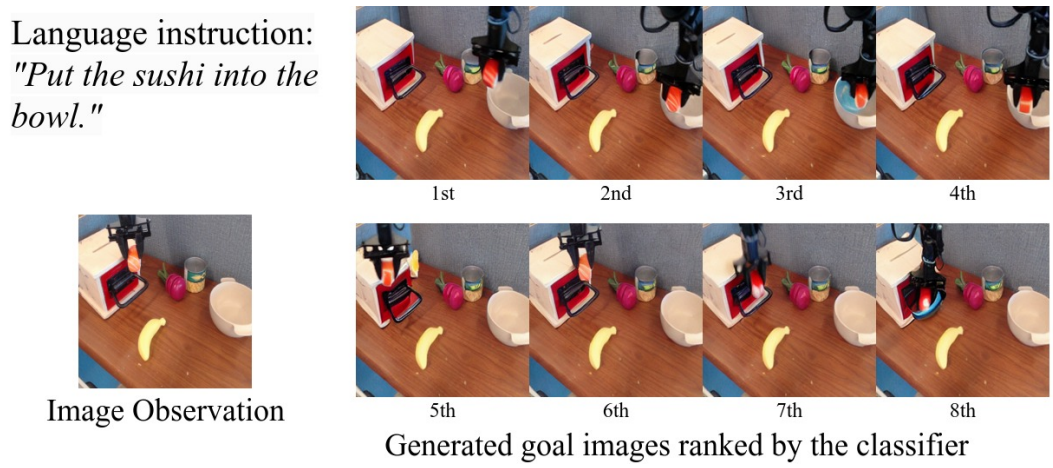
C.1 Classifier rankings

We show examples of how the classifier network ranks generated goal images on tasks from Scene D of our physical experimental domain. Figures 5a, 5b, 5c show examples of the classifier correctly ranking the generated goal images (highly ranked images correspond to making progress towards correctly completing the language instruction), while fig. 5d shows an example of the classifier erroneously giving high rankings to goal images that do not make progress towards completing the language instruction. Note that while the classifier scores can be close across various goal images, so long as the relative ranking of the generated goal images is correct, then incorrect subgoal images will be rejected and correct subgoal images will be passed to the low-level policy.

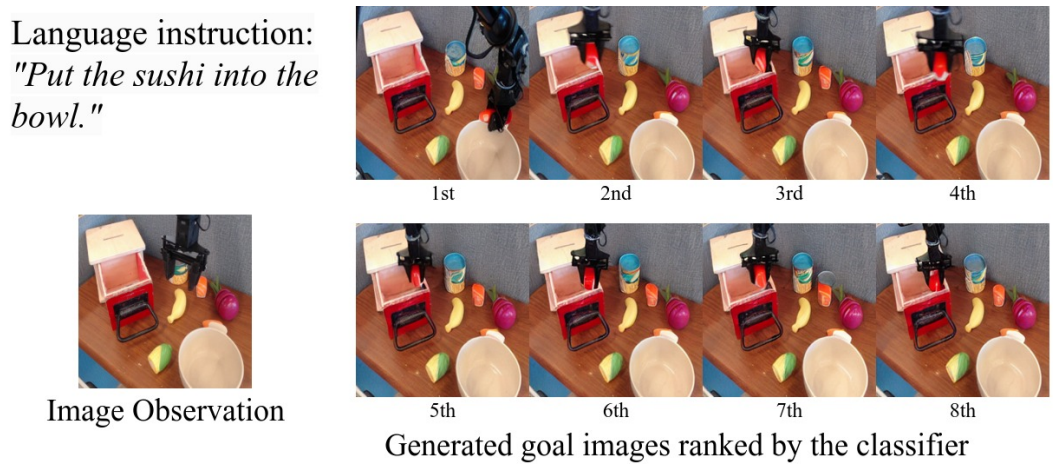
Figure 5: Classifier ranking examples Examples of the classifier network rankings on 8 generated candidate subgoals given an observation from Scene D of the physical experiments and a language instruction. Note that during Video Glue inference, only the first-ranked subgoal is passed to the low-level policy.



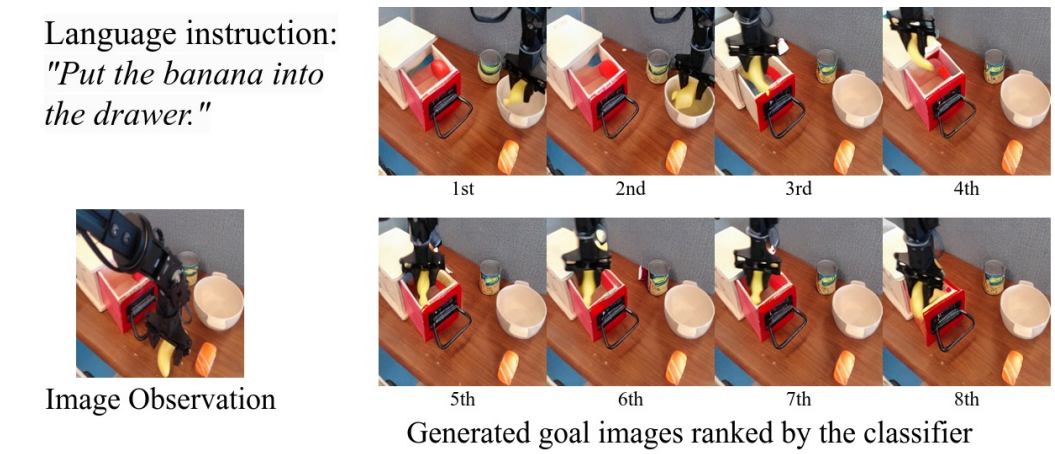
(a) Correct Example of Classifier Filtering The classifier correctly ranks the subgoal images where the robot is grasping the sushi higher than the subgoal images where the robot is grasping the drawer handle.



(b) Correct Example of Classifier Filtering The classifier correctly ranks the subgoal images where the robot moves to place the grasped sushi into the bowl higher than the subgoal images where the robot moves its gripper towards the drawer handle. It ranks the subgoal image with the hallucinated blue bowl-like artifact last.



(c) Correct Example of Classifier Filtering The classifier correctly ranks the subgoal image highest that shows the robot completing the correct task – only a single generated subgoal image shows the robot placing the sushi into the bowl, while all other generated subgoal images show the robot placing the sushi into the drawer.



(d) The classifier incorrectly ranks the subgoal images higher where the robot is placing the banana into the bowl than it ranks the subgoal images where the robot is placing the banana into the drawer. This could be due to there being a strong bias for placing objects in bowls in the Bridge V2 training data.

C.2 Trajectory Visualizations

We show examples of rollouts of Video Glue (SuSIE) on our physical experiment set up. These examples showcase when Video Glue successfully filters out off-task subgoal images (Figure 6a), as well as an instance of when Video Glue nearly causes a failure (Figure 6b).

C.3 Qualitative Analysis of Augmentation De-synchronization

We see that when applying augmentation de-synchronization, the number of failures due to low-level policy errors (missed grasps, dropping held objects, etc.) decreases, indicating that augmentation de-synchronization is important for the low-level policy to be able to correctly interpret and follow the subgoal images generated by the video prediction model. This is particularly important in domains where there is a large visual generalization gap between the training data and the evaluation tasks. For example, in the CALVIN benchmark, the colors and shapes of objects differ between the training and evaluation scenes. This difference causes the subgoals generated by the video prediction model to often contain objects with incorrect shapes and colors (Figure 7). Augmentation de-synchronization seems to be critical to allowing the low-level policy to be robust to these hallucinations and artifacts.

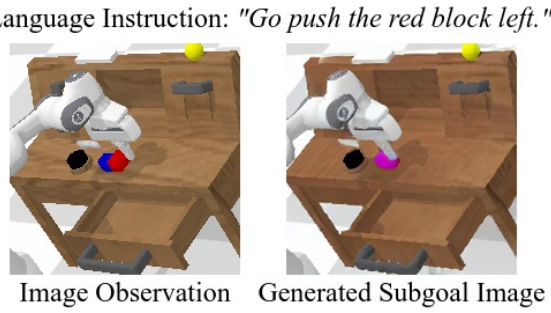
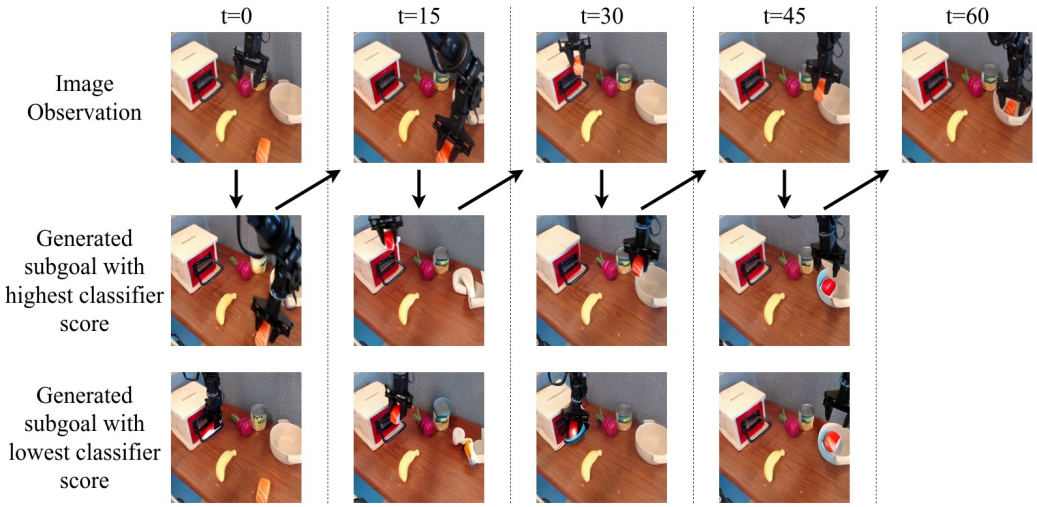


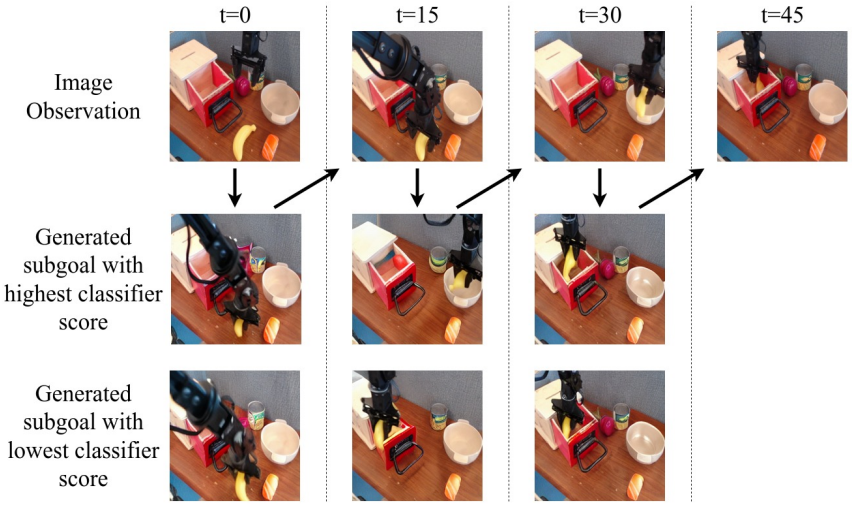
Figure 7: Generated Subgoal Image on CALVIN A subgoal image generated by the SuSIE video model on the unseen environment D of the CALVIN benchmark. The colors and shapes of objects are different in each of the four CALVIN environments, and since the model was not trained on data from environment D, it often generates images with incorrect shapes and colors. Augmentation de-synchronization is important for the low-level policy and classifier to be able to handle these mismatches between image observations and corresponding generated subgoal images.

Figure 6: Video Glue (SuSIE) Trajectory Visualization Visualization of a rollout of Video Glue (SuSIE) on Scene D in the physical experiments set up. The top row shows the current image observation at every timestep at which the video prediction model is queried. The second and third rows show the highest and lowest ranked generated subgoal images out of the 8 generated subgoal images, as ranked by the classifier. Note that during Video Glue inference, only the first-ranked subgoal is passed to the low-level policy.



Language instruction: *"Put the sushi into the bowl."*

(a) "Put the sushi into the bowl." This rollout shows two examples of the classifier filtering preventing the policy from going off-task: at $t = 0$, the lowest ranked generated subgoal shows the gripper grasping the drawer handle instead of moving to grasp the sushi; at $t = 30$, the lowest ranked generated subgoal shows the gripper moving towards the drawer handle instead of towards placing the sushi into the bowl. Note the hallucinated objects and artifacts visible in the goal images at $t = 15, 30, 45$. Augmentation de-synchronization helps to make the low-level policy and classifier robust to hallucinated artifacts such as these.



Language instruction: *"Put the banana into the drawer."*

(b) "Put the banana into the drawer." In this rollout, classifier filtering fails and causes a near-miss. At $t = 15$, the classifier ranks a subgoal image highest that shows the robot placing the banana into the bowl instead of the drawer. However, at $t = 30$, when the robot reaches the state specified by this subgoal image, the subsequent generated subgoals all show the robot correctly placing the banana into the drawer. Although, as in this example, the classifier network can occasionally rank incorrect subgoal images higher than correct subgoal images, such errors occur infrequently as Video Glue (SuSIE/UniPI) outperforms base-SuSIE/UniPi across all of our physical and simulated experiments.

D Additional Ablation Experiments

Method	Tasks completed in a row					
	1	2	3	4	5	Avg. Len.
SuSIE	89.8%	75.0%	57.5%	41.8%	29.8%	2.94
Video Glue (SuSIE) -{Subgoal Filtering}	95.2%	84.0%	69.5%	56.0%	46.2%	3.51
Video Glue (SuSIE) -{Aug De-sync}	88.5%	75.5%	56.2%	43.0%	32.5%	2.96
Video Glue (SuSIE) -{Aug De-sync on policy}	91.5%	74.2%	56.0%	41.2%	29.8%	2.93
Video Glue (SuSIE) -{Aug De-sync on classifier}	95.0%	86.2%	70.0%	57.8%	47.0%	3.56
Video Glue (SuSIE)	**95.2%**	**88.5%**	**73.2%**	**62.5%**	**49.8%**	**3.69**

Table 4: Effect of Augmentation Desynchronization in Video Glue (SuSIE) Success rates on the validation tasks from the D environment of the CALVIN Challenge averaged across 4 random seeds. Removing the augmentation desynchronization from only the low-level policy results in similar performance to base-SuSIE and Video Glue (SuSIE) with the augmentation desynchronization removed from both the low-level policy and the classifier. This suggests that the low-level policy performance of SuSIE without augmentation desynchronization is a significant bottleneck for SuSIE–even when selecting better subgoals via the use of filtering, performance cannot increase if the low-level policy cannot reliably reach those goals. Conversely, removing the augmentation desynchronization from only the classifier results in similar performance to Video Glue (SuSIE) without subgoal filtering. This suggests that, like the low-level policy, augmentation desynchronization is important for the classifier network to correctly perform its function in Video Glue (SuSIE).

Method	Tasks completed in a row					
	1	2	3	4	5	Avg. Len.
UniPi	22.0%	2.5%	0.5%	0.2%	0.0%	0.25
Video Glue (UniPi) -{Subgoal Filtering}	22.0%	4.0%	0.2%	0.0%	0.0%	0.26
Video Glue (UniPi) -{Aug De-sync}	43.8%	17.5%	4.2%	0.5%	0.0%	0.66
Video Glue (UniPi) -{Aug De-sync on policy}	39.0%	16.5%	6.5%	1.0%	0.0%	0.63
Video Glue (UniPi) -{Aug De-sync on classifier}	**48.0%**	**20.0%**	**3.5%**	**0.5%**	**0.0%**	**0.72**
Video Glue (UniPi)	40.2%	13.5%	5.8%	0.5%	0.0%	0.60

Table 5: Effect of Augmentation Desynchronization in Video Glue (UniPi) Success rates on the validation tasks from environment D of the CALVIN Challenge averaged across 4 random seeds. Removing the augmentation desynchronization from only the low-level policy results in similar performance to Video Glue (UniPi) with the augmentation desynchronization removed from both the low-level policy and the classifier. This suggests that, with UniPi, subgoal filtering has a much larger influence on performance than augmentation de-synchronization. Surprisingly, removing the augmentation desynchronization from only the classifier results in better performance than the other methods, perhaps suggesting that subgoals generated by the UniPi model have a high degree of diversity and incorrect subgoals can be easily identified and filtered out by the classifier.

Method	Tasks completed in a row					
	1	2	3	4	5	Avg. Len.
Video Glue (SuSIE) - 4 samples	95.2%	86.0%	71.2%	60.5%	50.0%	3.63
Video Glue (SuSIE) - 8 samples	**95.2%**	**88.5%**	**73.2%**	**62.5%**	**49.8%**	**3.69**
Video Glue (SuSIE) - 16 samples	95.0%	86.5%	72.8%	60.8%	48.0%	3.63

Table 6: Effect of Number of Candidate Goal Images Sampled in Video Glue (SuSIE) Success rates on the validation tasks from environment D of the CALVIN Challenge when using Video Glue (SuSIE) when using 4, 8, or 16 candidate goal images with classifier filtering. Results are averaged across 4 random seeds. Results are similar across all numbers of samples, with 8 samples performing the best by a slight margin.