GHIL-Glue: Hierarchical Control with Filtered Subgoal Images

Anonymous Author(s)
Affiliation
Address
email

Abstract: Image and video generative models that are pre-trained on Internet-scale data can increase the generalization capacity of robot learning systems. These models can function as high-level planners, generating intermediate subgoals for low-level goal-conditioned policies to reach. However, the performance of these systems can be bottlenecked by the interface between generative models and low-level controllers. Generative models may predict photorealistic yet physically infeasible frames. Low-level policies may also be sensitive to subtle visual artifacts in generated goal images. This paper addresses these facets of generalization, providing an interface to "glue together" language-conditioned image or video prediction models with low-level goal-conditioned policies. Our method, Generative Hierarchical Imitation Learning-Glue (GHIL-Glue), filters out subgoals that do not lead to task progress and improves the robustness of goal-conditioned policies to generated subgoals with harmful visual artifacts. GHIL-Glue achieves a new state-of-the-art on the CALVIN simulation benchmark for policies using observations from a single RGB camera. GHIL-Glue also outperforms other generalist robot policies across 3/4 language-conditioned manipulation tasks testing zero-shot generalization on a physical robot. Additional details are available at https://generative-hierarchical-glue.github.io.

Keywords: Hierarchical Imitation Learning, Image Generation, Video Prediction

1 Introduction

As Internet-scale foundation models achieve success in computer vision and natural language processing, a central question arises for robot learning: how can Internet-scale models enable embodied behavior generalization? While one approach is to collect increasingly large action-labeled robot manipulation training datasets [1, 2, 3], video datasets (without actions) from the Internet are vastly larger. However, while videos may be useful for inferring the steps in a task, such as how the objects should be moved, or which parts of an object to manipulate (e.g., grabbing a cup by the handle), they are less useful for learning details about low-level control. For example, it is difficult to infer the actions for controlling a robot's fingers from videos of humans performing manipulation tasks. One promising solution to this challenge is to employ a hierarchical approach [4, 5]: infer high-level subgoal images using models trained on Internet-scale videos, and then fill in the fine-grained motions with low-level policies trained on robot data (see appendix A for a discussion of related work).

While this general approach has seen success in prior robotic manipulation work [6, 4, 7, 5, 8, 9], the interface between the high-level planner generating subgoals and the low-level policy that must reach these subgoals can be brittle. First, generative models may occasionally sample subgoals that do not progress towards completing a given language instruction. If one such "off-task" subgoal is followed, it can have a compounding errors effect, leading to subsequent subgoals being increasingly "off-task." Second, even if the generated subgoals lead to task progress, they can contain subtle visual artifacts that degrade the performance of a naively trained low-level policy.

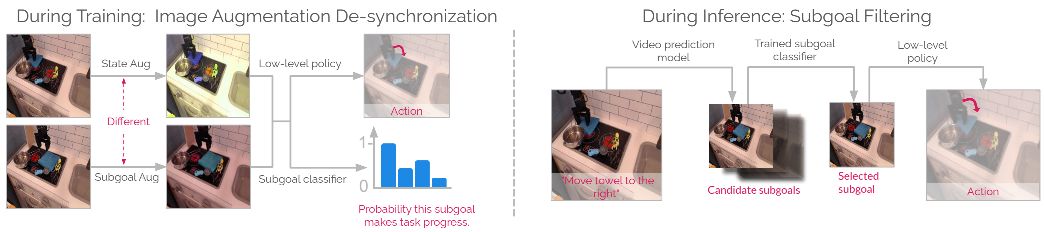


Figure 1: GHIL-Glue. We consider language-conditioned image and video prediction models that can generate multiple subgoals. GHIL-Glue has two components: augmentation de-synchronization (top) and subgoal filtering (bottom). Subgoal filtering: We train a classifier to identify which subgoal is most likely to progress towards completing the language instruction. This subgoal and the image observation are then passed to the low-level policy to choose a robot action. Augmentation de-synchronization: The distribution shift between subgoals sampled from the robot dataset during training and those sampled from the generative model during inference can degrade low-level policy and subgoal classifier performance. To robustify the low-level policy and subgoal classifier to artifacts in generated subgoals, we explicitly de-synchronize the image-augmentations applied to the current state (State Aug) and the sampled goal (Subgoal Aug).

To address these issues, we propose Generative Hierarchical Imitation Learning-Glue (GHIL-Glue) (fig. 1), a method to *robustly* "glue" together image or video generative models to a low-level robotic control policy. **First**, we filter out "off-task" subgoals that are physically inconsistent with the commanded language instruction. We do this by training a subgoal classifier to predict the likelihood of the transition between the current state and a given subgoal resulting in progress towards completing the provided language instruction. We then sample a number of candidate subgoals from the generative model and choose the subgoal with the highest classifier ranking. **Second**, we identify a simple yet non-obvious data augmentation practice to robustify the low-level policy and subgoal classifier to visual artifacts in the generated subgoals. While image augmentations are ubiquitous in robot learning methods, our key finding is that the standard way of applying image augmentations does not make low-level policies robust to visual artifacts in generated subgoal images. Experiments on the CALVIN [10] simulation benchmark and four language-conditioned tasks on the Bridge V2 physical robot platform [11] suggest that GHIL-Glue improves upon prior SOTA methods for zero-shot generalization while adding minimal additional algorithmic complexity.

2 GHIL-Glue

2.1 Subgoal Filtering

The image and video generative models we consider are first pre-trained on general Internet-scale image and video data, and then fine-tuned on a modest amount of robot data (see appendix B for a detailed description of the problem setting we consider). A common failure mode we observe across different models is that, while executing a task, the model begins to go "off-task," generating subgoals that are consistent with the current image observation but that do not progress towards completing the language instruction l. We hypothesize that this is due to the distribution shift between the Internet image and video pre-training data and the robot data they are fine-tuned on.

To address this challenge, we train a subgoal classifier $f_\theta(s, g, l)$ on a language-conditioned dataset of trajectories $\mathcal{D}_l$ 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. During training, we sample positive examples of state-goal transitions for l from the set of trajectories that successfully complete the instruction. We construct negative examples 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, and **3) Reverse Direction**: (g, s, l), where the order of the current image observation and the subgoal image have been switched. We refer to this dataset of negative examples constructed from $\mathcal{D}_l$ as $\mathcal{D}_l^-$. We then train the subgoal classifier by minimizing the binary cross entropy loss between the positive examples and the constructed negative examples:

$\mathcal{J}(\theta) = \mathbb{E}_{(s,g,l)\sim\mathcal{D}_l}\left[\log\left(f_\theta(s,g,l)\right)\right] + \mathbb{E}_{(s^-,g^-,l^-)\sim\mathcal{D}_l^-}\left[\log\left(1 - f_\theta(s^-,g^-,l^-)\right)\right]$. At inference, given a set of K subgoals predicted by the image or video model, GHIL-Glue selects the subgoal with the highest classifier ranking to the low-level policy for conditioning.

2.2 Image Augmentation De-Synchronization

For both the low-level goal-conditioned policy and the subgoal classifier, each training sample includes two images: the current state s and the corresponding goal g. Applying image augmentation procedures during training is a standard approach in image-based robot learning methods [12] to improve the robustness of learned models to distribution shifts between their training and evaluation domains. Standard practice is to sample augmentation parameters $\hat{\phi}$ and apply them to all images in a given training sample [4, 13], which corresponds to applying the same $\hat{\phi}$ to both s and g. In a non-hierarchical policy setting, this makes sense, because at inference time s and g will both be sampled from the camera observations of the current environment instantiation. However, when using an image or video prediction model for subgoal generation, at inference time the observations will come from the environment, but the goals will be generated by the image or video prediction model. There will often be differences in the visual artifacts between a camera observation s and the corresponding generated subgoal image g, such as differences in color, contrast, blurriness, and the shapes of objects, which can degrade the performance of low-level policies and subgoal classifiers.

To encourage robustness to this distribution shift, we sample separate augmentation parameters for s and g, denoted by $\hat{\phi}_s$ and $\hat{\phi}_g$ (i.e., we de-synchronize the image augmentations applied to s and g). Concretely, for each s and g pair sampled during training, a different random crop, brightness, contrast, saturation, and hue shift are applied to s than are applied to g. This forces the low-level policy and the subgoal classifier to be robust to differences in visual artifacts between s and g. See appendix C for additional discussion of image augmentation de-synchronization.

3 Experiments

3.1 Experimental Domains

Simulation Experiment Setup: Simulation experiments are performed in the CALVIN [10] benchmark, which focuses on long-horizon language-conditioned robot manipulation. We follow the same protocol as in [4], and train on data from three environments (A, B, and C) and test policies on a fully unseen environment (D). The held-out environment (D) contains unseen desk and object colors, positions, and shapes. See appendix D for a visualization of the CALVIN environment.

Physical Experiment Setup: Physical experiments are performed with the Bridge V2 [11] experiment setup with a WidowX250 robot. We use the same datasets as in [4] for training both the high-level image 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 [4], we use a filtered version of the Something-Something V2 dataset [14] with the same filtering scheme as in [4] (resulting in 75K video clips) as our video-only dataset $\mathcal{D}_l$. We test our policies on four tasks on four different cluttered table top scenes (fig. 2) on the Bridge V2 physical robot platform. These environments require generalizing to novel scenes, with novel objects, and with novel language commands that are not seen in the Bridge V2 dataset. See appendix D for visualizations of the evaluation set-up.

3.2 Comparison Algorithms

We study the impact of applying GHIL-Glue to two SOTA hierarchical imitation learning algorithms: SuSIE [4] and UniPi [5]. We use either 4 or 8 candidate subgoals for subgoal filtering (see appendix J for details). We also compare GHIL-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 (section 2.1) and de-synchronizing augmentations (section 2.2). For physical experiments, we additionally compare to OpenVLA [15], which is trained on the Open X-Embodiment dataset [2] (which includes the Bridge V2 dataset). See appendix E for a detailed description of each of these algorithms.

3.3 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 [4]	89.8%	75.0%	57.5%	41.8%	29.8%	2.94
GHIL-Glue (SuSIE) - Aug De-sync Only	95.2%	84.0%	69.5%	56.0%	46.2%	3.51
GHIL-Glue (SuSIE) - Subgoal Filtering Only	88.5%	75.5%	56.2%	43.0%	32.5%	2.96
GHIL-Glue (SuSIE)	**95.2%**	**88.5%**	**73.2%**	**62.5%**	**49.8%**	**3.69**
UniPi [5]	56.8%	28.3%	12.0%	3.5%	1.5%	1.02
GHIL-Glue (UniPi) - Aug De-sync Only	60.2%	29.5%	12.5%	5.5%	1.8%	1.1
GHIL-Glue (UniPi) - Subgoal Filtering Only	69.5%	40.0%	15.8%	6.5%	4.2%	1.36
GHIL-Glue (UniPi)	75.2%	44.8%	19.7%	11.2%	5.5%	1.56

Table 1: CALVIN: Simulation Results. Success rates on the validation tasks from the held-out D environment of the CALVIN zero-shot generalization challenge averaged across 4 random seeds. Applying GHIL-Glue to SuSIE and UniPi significantly improves performance over their respective base methods. GHIL-Glue (SuSIE) significantly outperforms all other methods, achieving a new state-of-the-art on the CALVIN benchmark for policies using observations from a single RGB camera.

	Task	OpenVLA [15]	SuSIE [4]	GHIL-Glue (SuSIE)
Scene A	Put Sushi On Towel	22/30	19/30	**28/30**
Scene B	Put Red Bell Pepper in Bowl	14/30	12/30	**16/30**
Scene C	Open Drawer	**23/30**	19/30	22/30
Scene D	Put Sushi in Bowl	15/30	15/30	**18/30**

Table 2: Bridge V2 Physical Experiments Results. Success rates across four tasks on four physical robot scenes (pictured in fig. 2) that test zero-shot generalization to novel objects, novel language commands, and novel scene configurations. GHIL-Glue applied to SuSIE outperforms SuSIE across all tasks and outperforms OpenVLA on 3 out of 4 tasks.

Simulation Experiments: We present results on the CALVIN benchmark in table 1. Applying GHIL-Glue yields significant improvements for SuSIE and UniPi, increasing the average successful task sequence length from **2.94** to **3.69** for SuSIE and from **1.02** to **1.56** for UniPi. **GHIL-Glue (SuSIE) achieves a new SOTA on CALVIN** for policies that use single RGB camera observations. See appendix F for additional discussion these of results.

Physical Experiments: We present results (table 2) comparing GHIL-Glue (SuSIE) to OpenVLA and SuSIE across four environments on the Bridge V2 robot platform that require interacting with a number of objects on a cluttered table (fig. 2). GHIL-Glue applied to SuSIE outperforms SuSIE across all tasks and outperforms OpenVLA, a 7-billion parameter SOTA VLA, on 3 out of 4 tasks. Significantly, the baseline SuSIE implementation does not outperform OpenVLA on a single task, whereas **GHIL-Glue (SuSIE) outperforms OpenVLA on 3 out of 4 tasks**, demonstrating that hierarchical goal conditioned architectures with well-tuned interfaces between the high and low-level policies can outperform SOTA VLA methods on zero-shot generalization tasks. See appendix F for additional discussion of results and appendix I for qualitative analysis of success and failure cases.

4 Conclusion

We present GHIL-Glue, a method for better aligning image and video prediction models and low-level control policies for hierarchical imitation learning. Our key insight is that while image and video foundation models can generate highly realistic subgoals for goal-conditioned policy learning, when generalizing to novel environments, the generated images are prone to containing visual artifacts and can be inconsistent with the task the robot is commanded to perform. GHIL-Glue provides two simple ideas to address these challenges, significantly improving zero-shot generalization performance over prior work in the CALVIN simulation benchmark and in physical experiments.

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] 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.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[16] 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.

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Conference on Robot Learning*, 2024.

[24] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning*, 2024.

[25] 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.

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

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

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

[46] 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.

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

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

[49] 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.

[50] 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.

[51] 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.

[52] 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.

[53] 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.

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

[55] 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.

[56] 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.

[57] 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.

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

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

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

[61] 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.

[62] 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.

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

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

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

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

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

[72] 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.

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

[74] 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.

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

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

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

[78] 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.

[79] 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.

[80] 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.

[81] 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.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

[87] J. Xing, M. Xia, Y. Zhang, H. Chen, W. Yu, H. Liu, X. Wang, T.-T. Wong, and Y. Shan. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023.

[88] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[89] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[90] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[91] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[92] 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 Related work

Generative Models for Robotic Control: Prior works have explored diverse ways to leverage generative models, such as diffusion models [16, 17] and Transformers [18], for robotic control. They have employed highly expressive generative models, potentially pre-trained on Internet-scale data, for low-level control [19, 20, 21, 22, 23, 24], data augmentation [25, 26, 27], object detection [28, 29], semantic planning [30, 31, 32, 33, 34], and visual planning [6, 4, 7, 5, 8, 9]. 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 [6, 4, 7, 5, 8, 9]. 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, this configuration leads to failures due to the misalignment of the generative models and the low-level policies that control the robot behavior, as shown in our experiments (section 3).

Rejection Sampling: One of our key ideas in this paper is based on rejection sampling, where we sample multiple subgoal proposals from an image or video prediction model and pick the best one based on a learned subgoal classifier. 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) [35, 36, 37, 38], response verification in natural language processing [39, 40, 41], and planning and exploration in robotics [42, 32, 33, 43, 44]. Previous works in robotics have proposed several ways to filter out infeasible plans generated by pre-trained foundation models [42, 32, 33, 43, 45]. Unlike these works, we focus on filtering visual subgoals instead of language plans [32, 43, 45], and do not involve any planning procedures [33] or structural knowledge [42]. While the subgoal classifier we train resembles the classifier from [46], our classifier differs in two key ways. First, we use our classifier to filter out "off-task" subgoals, whereas the classifier in [46] is used as a reward function for training downstream policies. Second, the classifier from [46] is conditioned on the initial state s_0 and the current state s, whereas our classifier is conditioned on the current state s and a generated subgoal g.

Goal-Conditioned Policy Learning: Our method is broadly related to goal-conditioned policy learning [47, 48, 49], language-conditioned policy learning [50, 51, 52, 53, 54], and hierarchical control [4, 5, 55, 56, 57, 58]. Most prior works in hierarchical policy learning either train a high-level policy from scratch that produces subgoals or latent skills [59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 56] or employ subgoal planning [73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 69, 85]. 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) image or video prediction model and pick the best one based on a trained subgoal classifier. Among hierarchical policy methods, perhaps the closest work to ours is IRIS [55], 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 differs from IRIS in that we do not assume access to a reward function in order to train a value function. Our classifier is trained on trajectories consisting only of images and language descriptions.

Diffusion Model Guidance: The generative models we consider in our paper [86, 87] are diffusion-based models trained using classifier-free guidance (CfG) [88]. Although we use a large value for the language-prompt guidance parameter at inference in our experiments, we find that producing "off-task" subgoals is still a common failure mode that is not solved by increasing this parameter alone.

Classifier guidance [16, 89, 90] is also a plausible alternative to rejection sampling, but there are some practical challenges in training a subgoal classifier for this purpose. First, the diffusion models we consider use latent diffusion [91], and therefore would require training the subgoal classifier to operate in the latent space of the diffusion model. Second, the subgoal classifier would need to be trained on noised data in order to guide the diffusion denoising process of the generative model. Nevertheless, classifier guidance is a potentially appealing direction for future work.

B Preliminaries

We consider the same problem setting as [4], 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 [4]: (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. This can be trained on $\mathcal{D}_l$ and $\mathcal{D}_{l,a}$.
2. **a low-level goal-reaching policy** that chooses actions to reach generated subgoals. This can be trained on $\mathcal{D}_a$ and/or $\mathcal{D}_{l,a}$.

Our contribution is a set of approaches to robustify the interface between these two modules.

While GHIL-Glue can be applied to any hierarchical imitation learning method consisting of the two components mentioned above, in this work we apply GHIL-Glue to two specific algorithms: (1) UniPi [5], in which a high-level model generates a subgoal video, and a low-level inverse-dynamics model predicts the actions needed to "connect" the images in the video, and (2) SuSIE [4], in which a high-level model generates a subgoal image by "editing" the current image observation, and a goal-conditioned policy predicts actions to achieve the subgoal image. We define subgoals, $g \in \mathcal{G}$, as video or image samples from the high-level models used in these algorithms.

C Additional Discussion of Image Augmentation De-Synchronization

Generated subgoals can contain visual artifacts that degrade the performance of both the low-level control policy and the subgoal classifier. This performance degradation results from the distribution shift between the subgoal images seen by the policy during training, which come from the robot dataset, and the subgoal images seen during inference, which come from the generative model. Ideally, the low-level policy and subgoal classifier would be trained on the same distribution of *generated* subgoal images that they will see at inference time. However, due to the high degree of variance in sampling images from a generative model, there is not a clear way to obtain generated subgoal images that match the actual future states reached in trajectories in the training data. To address this issue, we identify a simple yet non-obvious data augmentation practice to train the low-level policy and subgoal classifier on goals from the robot dataset while also robustifying them to visual artifacts in generated subgoals.

Applying image augmentation procedures such as random cropping or color jitter during training is a standard approach in image-based robot learning methods [12] 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 make a model robust to any augmentation in the space Φ.

For both the low-level goal-conditioned policy and the subgoal classifier, 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 [4, 13], which corresponds to applying the same $\hat{\phi}$ to both s and g. In a non-hierarchical policy setting, this makes sense, because at inference time s and g will both be sampled from the camera observations of the current environment instantiation. However, when using an image or video prediction model for subgoal generation, at inference time the low-level policy and subgoal classifier will see states from the camera observations, but the goals will be generated by the image or video prediction model. There will often be differences in the visual artifacts between a camera observation s and the corresponding generated subgoal image g, such as differences in color, contrast, blurriness, and the shapes of objects, which can degrade the performance of low-level policies and subgoal classifiers.

To encourage robustness to this distribution shift, we sample separate augmentation parameters for s and g, denoted by $\hat{\phi}_s$ and $\hat{\phi}_g$ (i.e., we de-synchronize the image augmentations applied to s and g). Random cropping, brightness shifts, contrast shifts, saturation shifts, and hue shifts comprise our space of augmentations. Concretely, for each s and g pair sampled during training, a different random crop, brightness, contrast, saturation, and hue shift are applied to s than are applied to g. This forces the low-level policy and the subgoal classifier to learn to make accurate predictions on (s, g) pairs that have differences in visual artifacts.

While image augmentations are ubiquitous in robot learning methods, our experiments show that the standard way of applying image augmentations for goal-conditioned policies and classifiers is deficient for the hierarchical policy methods that we consider. We also note that augmentation desynchronization is applied not only to the policy, but also to the subgoal classifier (section 2.1), which has a significant impact on overall performance (section 3).

D Experimental Domains

We study the degree to which GHIL-Glue improves existing hierarchical imitation learning algorithms across a number of tasks in simulation and physical experiments that assess zero-shot generalization. We evaluate our method on the CALVIN [10] simulation benchmark and the Bridge V2 [11] physical experiment setup with a WidowX250 robot. The experimental domains are visualized in fig. 2.

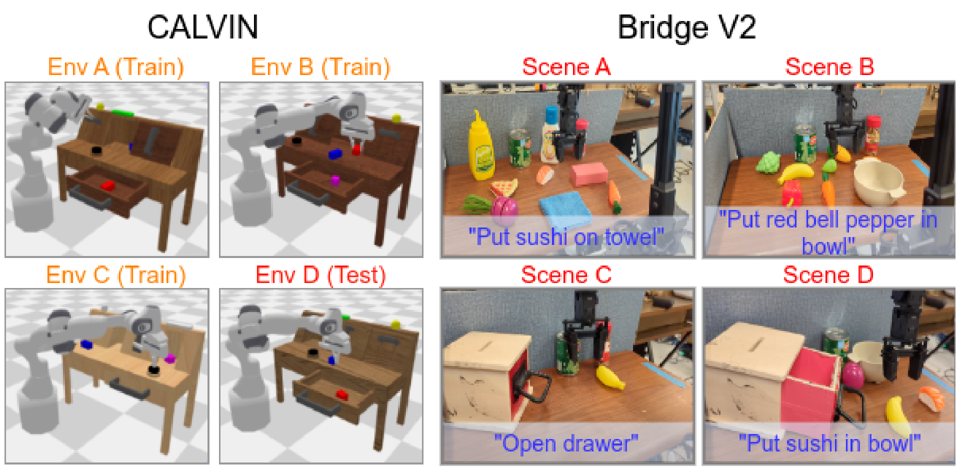


Figure 2: Experimental Domains. Simulation Environments (Left): Train/test environments in the CALVIN simulation benchmark. The environments each have different table textures, furniture positions, and initial configurations of the colored blocks. Each environment contains 34 tasks, each with an associated language instruction. To test zero-shot generalization, environment D is held out for evaluation. Physical Environments (Right): We consider four test scenes in the Bridge V2 robot platform with four total language instructions. To test zero-shot generalization, these test scenes contain novel objects, language commands, and object configurations not seen in the training data.

E Comparison Algorithms

A detailed description of the comparison algorithms referenced in section 3.2 is provided below:

1. **LCBC Diffusion Policy:** Low-level language-conditioned behavior cloning diffusion policy [20] trained only on robot trajectories with language annotations. We use the same implementation as in [4].

2. **OpenVLA [21]:** A SOTA language-conditioned vision-language-action model (VLA) trained on the Open X-Embodiment dataset [2] (which includes the entirety of the Bridge V2 dataset).

3. **SuSIE [4]:** A method which fine-tunes InstructPix2Pix [86], 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 [5]:** A method which fine-tunes 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 by fine-tuning the video model from [87].

5. **GHIL-Glue (SuSIE / UniPi):** GHIL-Glue applied on top of either SuSIE or UniPi. For all experiments we implement the subgoal filtering step by sampling four to eight 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.

6. **GHIL-Glue (SuSIE / UniPi) - Subgoal Filtering Only:** GHIL-Glue applied to SuSIE or UniPi using subgoal filtering but without augmentation de-synchronization.

7. **GHIL-Glue (SuSIE / UniPi) - Aug De-sync Only:** GHIL-Glue applied to SuSIE or UniPi using augmentation de-synchronization but without subgoal filtering.

F Discussion of Results

Simulation Experiments: We present results on the CALVIN benchmark in table 1. Applying GHIL-Glue yields significant performance increases for SuSIE and UniPi, increasing the average successful task sequence length from **2.94** to **3.69** for SuSIE and from **1.02** to **1.56** for UniPi. **GHIL-Glue (SuSIE) achieves a new SOTA on CALVIN for policies that use observations from a single RGB camera.** The two components of GHIL-Glue (subgoal filtering and image augmentation de-synchronization) improve performance when applied individually, but, when applied together, these components build on each other, leading to a performance increase greater than the sum of the individual benefits. Specifically, for SuSIE, image augmentation de-synchronization and subgoal filtering individually yield increases in sequence length of 0.56 and 0.02 respectively, whereas when applied together they yield an increase of 0.75. Similarly, for UniPi, the individual improvements yield increases in sequence length of 0.08 and 0.34 respectively, compared to an increase of 0.54 when applied together.

When applied alone, image augmentation de-synchronization increases the average successful task sequence length from 2.94 to 3.51 for SuSIE and from 1.02 to 1.1 for UniPi. We hypothesize that augmentation de-synchronization improves performance a large amount with SuSIE because its low-level policy is conditioned on a camera observation image s from the environment and a subgoal image g generated by the image model. When generalizing to the held-out test environment D, the SuSIE image model generates subgoal images with visual discrepancies from the camera observation images. In contrast, the UniPi video model predicts a sequence of frames as opposed to a single subgoal image. The UniPi low-level policy functions as an inverse dynamics model, choosing actions to link between the frames of the generated subgoal video, and is therefore conditioned on an s and g that both come from the predicted subgoal video.

When applied alone, subgoal filtering has a small effect on SuSIE, while on UniPi it increases the average successful task sequence length from 1.02 to 1.36. This suggests that unless the SuSIE low-level policy is made robust to visual artifacts in generated subgoals, simply selecting the most task relevant subgoal is insufficient to improve performance. As discussed previously, the SuSIE low-level policy is more sensitive to visual artifacts in generated subgoals than is the UniPi inverse dynamics model.

Physical Experiments: We present results (table 2) comparing GHIL-Glue (SuSIE) to OpenVLA and SuSIE across four environments on the Bridge V2 robot platform that require interacting with a number of objects on a cluttered table (fig. 2). These environments require generalizing to novel scenes, with novel objects, and with novel language commands that are not seen in the Bridge V2 dataset. GHIL-Glue applied to SuSIE outperforms SuSIE across all tasks and outperforms OpenVLA, a 7-billion parameter SOTA VLA, on 3 out of 4 tasks. Significantly, the baseline SuSIE implementation does not outperform OpenVLA on a single task, whereas GHIL-Glue (SuSIE) outperforms OpenVLA on 3 out of 4 tasks, demonstrating that hierarchical goal conditioned architectures with well-tuned interfaces between the high and low-level policies can outperform SOTA VLA methods on zero-shot generalization tasks.

G Classifier Training

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

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

where D_l 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, N generates negatives from D_l 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 [11], 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 [92] 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.

H 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 3 visualizes examples from the Bridge dataset before and after augmentations are applied:

Figure 3: 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.

I Qualitative Analysis

I.1 Effect of subgoal filtering

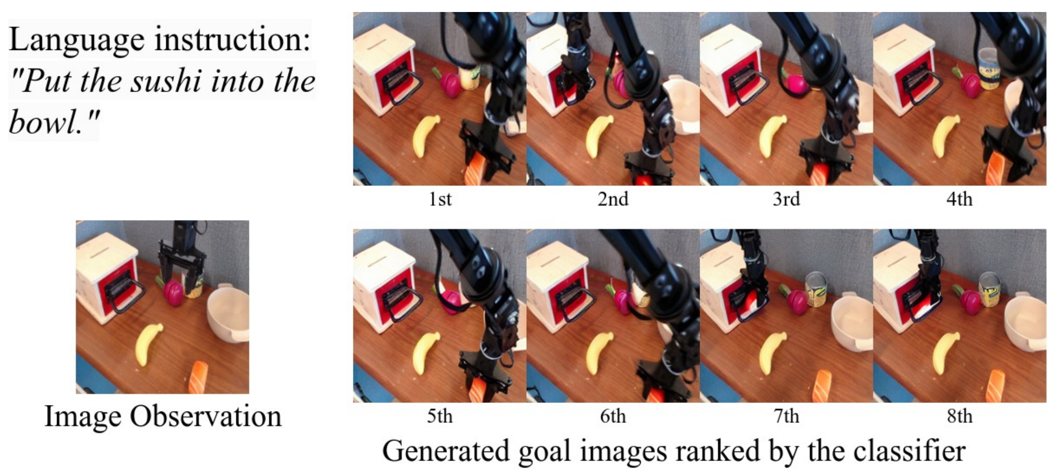
Although we use classifier-free guidance (CfG) [88] on the image or video generative model with respect to the language-prompt at inference in our experiments, we find that producing "off-task" subgoals is still a common failure mode that is not solved by increasing the guidance parameter alone. In fig. 4, we visualize how subgoal filtering can prevent "off-task" subgoals generated by the image or video model from being passed to the low-level control policy.

I.2 Classifier rankings

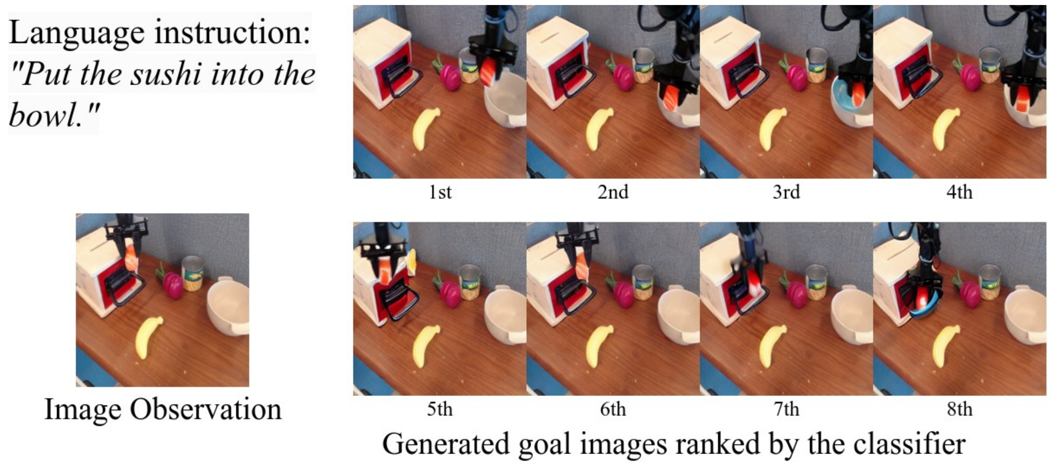
We show examples of how the classifier network ranks generated goal images on tasks from Scene D of our physical experimental domain. Figures 5a, 5b, 5c show examples of the classifier correctly ranking the generated goal images (highly ranked images correspond to making progress towards correctly completing 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 4: GHIL-Glue Subgoal Filtering. We visualize policy rollouts of SuSIE without subgoal filtering vs. GHIL-Glue SuSIE with subgoal filtering. We show the states reached every 20 timesteps (top row) and the corresponding predicted subgoals (bottom row). Without subgoal filtering, the subgoal at $t = 60$ is not consistent with making progress towards placing the pepper in the bowl, causing the robot to dither and drop the pepper. When subgoal filtering is used, the selected subgoals make iterative progress towards a successful task completion.

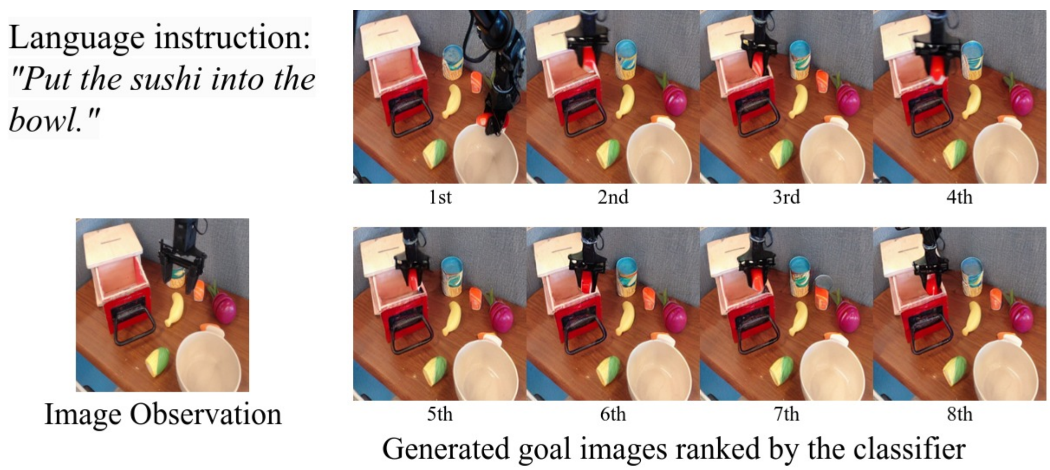
Figure 5: Classifier ranking examples Examples of the classifier network rankings on 8 generated candidate subgoals given an observation from Scene D of the physical experiments and a language instruction. Note that during GHIL-Glue inference, only the first-ranked subgoal is passed to the low-level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



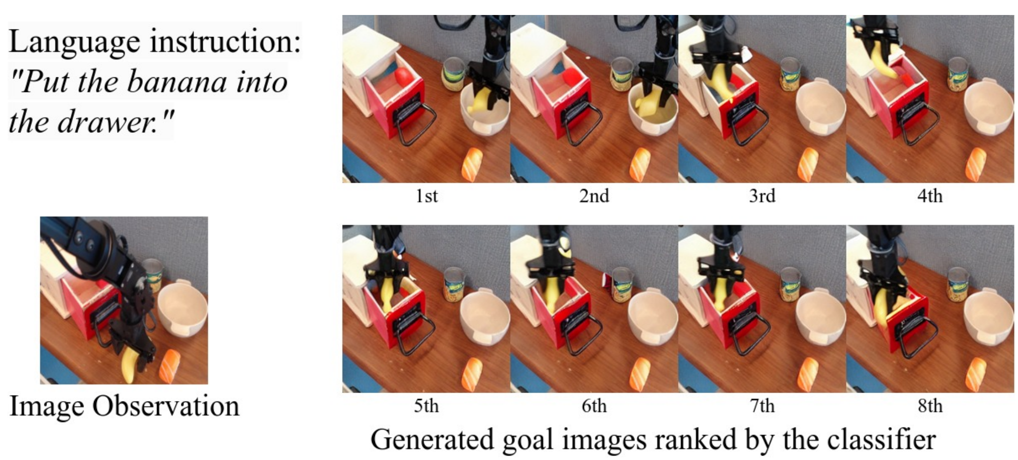
(a) Correct Example of Classifier Filtering The classifier correctly ranks the subgoal images where the robot is grasping the sushi higher than the subgoal images where the robot is grasping the drawer handle.



(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.

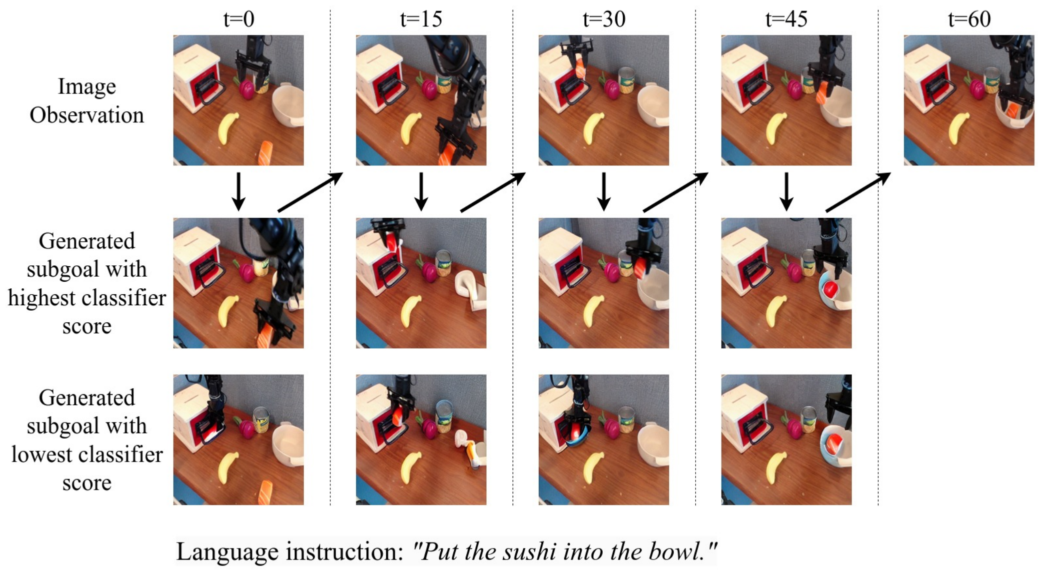


(d) The classifier incorrectly ranks the subgoal images higher where the robot is placing the banana into the bowl than it ranks the subgoal images where the robot is placing the banana into the drawer. This could be due to there being a strong bias for placing objects in bowls in the Bridge V2 training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

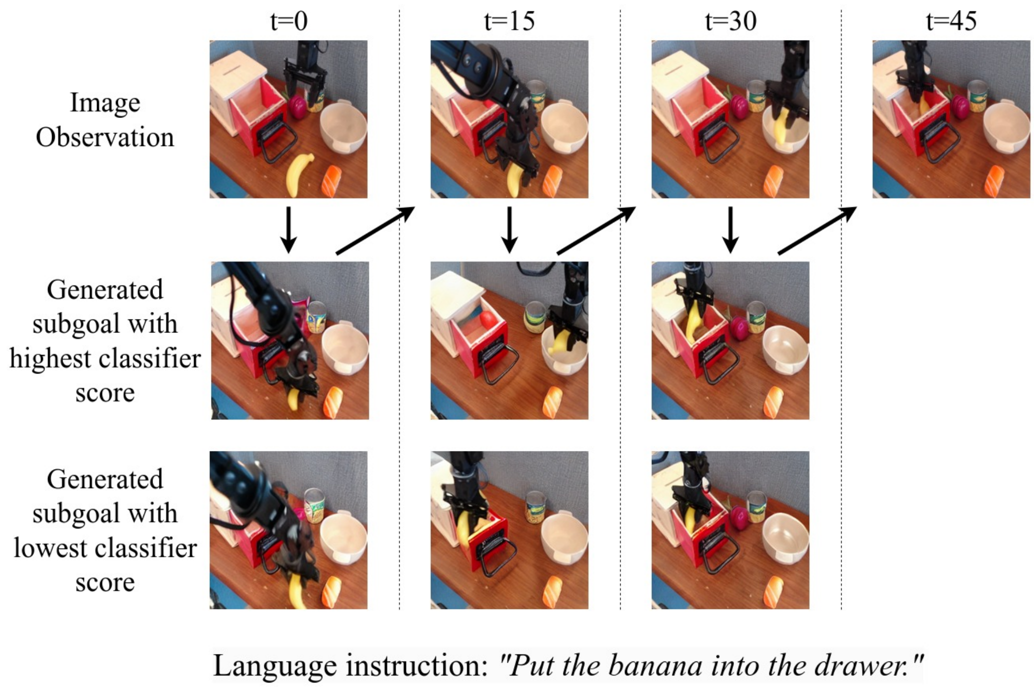
I.3 Trajectory Visualizations

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

Figure 6: GHIL-Glue (SuSIE) Trajectory Visualization Visualization of a rollout of GHIL-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 GHIL-Glue inference, only the first-ranked subgoal is passed to the low-level policy.



(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.



(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 GHIL-Glue (SuSIE/UniPI) outperforms base-SuSIE/UniPi across all of our physical and simulated experiments.

I.4 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.

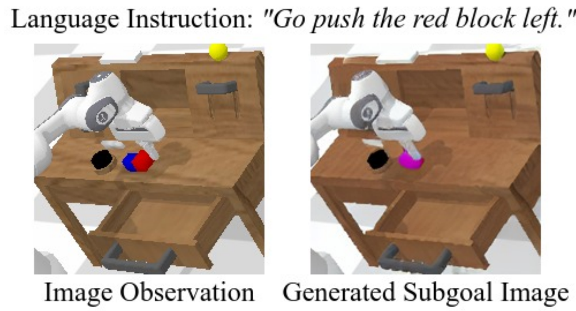


Figure 7: Generated Subgoal Image on CALVIN A subgoal image generated by the SuSIE video model on the unseen environment D of the CALVIN benchmark. The colors and shapes of objects are different in each of the four CALVIN environments, and since the model was not trained on data from environment D, it 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.

J Number of Candidate Subgoals

We conduct an ablation over the number of candidate subgoals used for subgoal filtering in GHIL-Glue (SuSIE) in the CALVIN benchmark. We find that GHIL-Glue (SuSIE) achieves similar performance whether 4, 8, or 16 candidate subgoals are used. In our main results (section 3.3), we report the performance of GHIL-Glue (SuSIE) on the CALVIN benchmark when using 8 candidate subgoals for filtering. For GHIL-Glue (UniPi) on the CALVIN benchmark, we use 4 candidate subgoals for filtering, due to the increased computation burden of generating video subgoals with the UniPi video model vs. generating image subgoals with the SuSIE image model. In our physical experiments, we run GHIL-Glue (SuSIE) using 4 candidate subgoals for filtering.

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

Table 3: Effect of Number of Candidate Goal Images Sampled in GHIL-Glue (SuSIE) Success rates on the validation tasks from environment D of the CALVIN Challenge when using GHIL-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.