Language Control Diffusion: Efficiently Scaling through Space, Time, and Tasks

Edwin Zhang[1] Yujie Lu[1] William Wang[1] Amy Zhang[2 3]

Abstract

Training generalist agents is difficult across several axes, requiring us to deal with high-dimensional inputs (space), long horizons (time), and multiple and new tasks. Recent advances with architectures have allowed for improved scaling along one or two of these dimensions, but are still prohibitive computationally. In this paper, we propose to address all three axes by leveraging **L**anguage to **C**ontrol **D**iffusion models as a hierarchical planner conditioned on language (LCD). We effectively and efficiently scale diffusion models for planning in extended temporal, state, and task dimensions to tackle long horizon control problems conditioned on natural language instructions. We compare LCD with other state-of-the-art models on the CALVIN language robotics benchmark and find that LCD outperforms other SOTA methods in multi task success rates while dramatically improving computational efficiency with a single task success rate (SR) of 88.7% against the previous best of 82.6%. We show that LCD can successfully leverage the unique strength of diffusion models to produce coherent long range plans while addressing their weakness at generating low-level details and control.

1. Introduction

Generalist agents are characterized by the ability to plan over long horizons, understand and respond to human feedback, and generalize to new tasks based on that feedback. Language conditioning is an intuitive way to achieve human-robot alignment for task specification, with built-in structure allowing for generalization to new tasks, as described by a new language command. There have been many recent developments to leverage language for robotics and downstream decision-making and control (Ahn et al., 2022; Li et al., 2022a; Mees et al., 2022a; Lynch & Sermanet, 2020a; Jiang et al., 2019; Colas et al., 2020). However, while language is useful for task specification and generalization, it will not necessarily help with planning over long horizons. One promising candidate that has emerged for long horizon planning is denoising diffusion models. Text to image diffusion models (Nichol et al., 2022; Ramesh et al., 2022; Ho et al., 2020; 2022c) have recently been able to successfully take advantage of large language models (LLMs) to generate incredibly detailed scenes. In addition, diffusion models have recently been proposed and successfully applied for low-dimensional, long-horizon planning and offline reinforcement learning (RL) (Janner et al., 2022; Wang et al., 2022b). In particular, Diffuser (Janner et al., 2022) has emerged as an especially promising planner with several properties suited for language conditioned control: flexibility in task specification, temporal compositionality, and ability to scale to long-horizon settings.

Current methods have a few pitfalls. Many existing language-conditioned control methods assume access to a high level discrete action space (e.g. switch on the stove, walk to the kitchen) provided by a lower level skill oracle (Ahn et al., 2022; Huang et al., 2022a; Jiang et al., 2022; Li et al., 2022a). The generative LM will typically decompose some high level language instruction into a set of predefined skills, which are then executed by a control policy or oracle. A fixed set of predefined skills may preclude the ability to generalize to novel environments and tasks. Further, one of the most important design considerations for these models is deciding what level of abstraction the communication protocol between the LM and the policy should take. A tradeoff arises from this question, where one must decide between the amount of reasoning to distribute over the LM versus the underlying policy. We can increase the burden on the LM by having it output text at a low level of abstraction that is extremely simple to follow, or we can increase the burden on the policy by having it execute text at a higher level of abstraction. Much prior work has been done on increasing the reasoning load of the language encoder by leveraging the capabilities of LLMs to generate planning code (Liang et al.,

[1]Department of Computer Science, University of California, Santa Barbara, USA [2]Meta AI, Menlo Park, USA [3]Department of Electrical and Computer Engineering, The University of Texas at Austin, USA. Correspondence to: Edwin Zhang <ete@cs.ucsb.edu>.

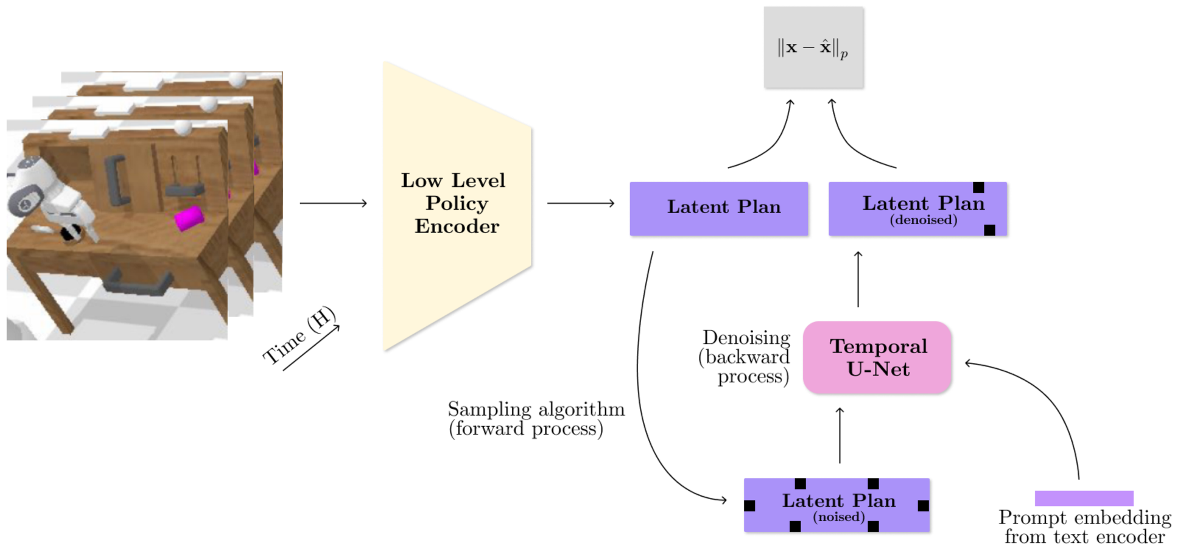

$\|\mathbf{x}-\hat{\mathbf{x}}\|_p$
Low Level Policy Encoder
Latent Plan
Latent Plan (denoised)
Time (H)
Denoising (backward process)
Temporal U-Net
Sampling algorithm (forward process)
Latent Plan (noised)
Prompt embedding from text encoder

Figure 1: An overview of our training pipeline. The low level policy encoder is used to encode a subsampled horizon of RGB observations into a lower dimensional latent space, which will be used later on as goals for a goal-conditioned policy. We then noise the plan according to a uniformly sampled timestep from the diffusion process' variance schedule, and train a Temporal U-Net to reverse this process conditioned on an natural language instruction from a frozen upstream large language model, effectively learning how to conditionally denoise the latent plan. To train the U-Net, one can simply use the p-norm between the predicted latent plan and the ground truth latent plan as the loss. We use $p = 1$ in practice following (Janner et al., 2022).

2022), calculate affordances of separate skills (Ahn et al., 2022), or perform chain of thought reasoning during rollout (Huang et al., 2022b), but scaling the low-level policy for interpreting higher level instruction has remained relatively underexplored.

Diffusion models are also computationally prohibitive to run in high-dimensional input spaces and tend to be inaccurate at low-level control. Specifically, Diffuser (Janner et al., 2022) does not work directly out of the box when scaling to pixels. This is perhaps unsurprising, as a high-dimensional Diffuser is effectively a text-to-video diffusion model, and even internet-scale video diffusion models have demonstrated a poor understanding of physics and temporal coherence over fine details (Ho et al., 2022a; Singer et al., 2022). This is due to the fact that the training objective for generative models has no notion of the underlying task, meaning they must model the entire scene with equal weighting. This includes potentially task-irrelevant details that may hinder or even prevent solving the actual control problem (Zhang et al., 2020; Wang et al., 2022a), which leads to catastrophic failure in control where fine details and precision are essential. Additionally, training a video diffusion model is generally computationally expensive and may take several days or weeks to train, which leaves such an approach out of reach to most researchers. This problem is exacerbated when considering that at inference time that multiple forward passes (often >20) are required for generating even a single state and action, meaning full text to video diffusion models are inefficient and likely impractical for the real-time sampling demands of robotics and control.

We address both of these issues by proposing the usage of a hierarchical diffusion model. By utilizing the representation of a goal-conditioned policy as a low level policy (LLP), we can effectively solve the representation issue by outputting states in a low-dimensional goal space directly into the LLP. This also allows us to arbitrarily scale the difficulty of the low-level policy learning problem by controlling the horizon length from which the goal state is set from the current state. This direction is promising, as it avoids defining the communication layer altogether and enables generating actions directly from high level text without a human-defined communication protocol in between. This hierarchical approach allows us to scale the diffusion model along three orthogonal axes:

- **Spatial dimension** through a low-dimensional representation that has been purposely optimized for control.
- **Time dimension** through a temporal abstraction enabled by utilizing a goal conditioned low level policy (LLP), as the LLP can use goal states several timesteps away from the current state.
- **Task dimension** through language, as the diffusion model acts as a powerful interpreter for plugging any large language model into control.

In addition, the entire pipeline is extremely fast and simple to train, as we utilize DDIM (Song et al., 2020a), a temporal abstraction on the horizon, as well as a low-dimensional representation for generation. We are able to achieve an average of 88.7% success rate across all tasks on the challenging CALVIN benchmark. Additionally, we elucidate where diffusion models for text-to-control work well and highlight their limitations. Finally, we explore learning the alignment and grounding between language and state-actions via evaluation of the task generalization capabilities of our hierarchical diffusion policy. In summary, we list our core contributions below:

- We propose an effective method for scaling diffusion models for multitask long horizon control from pixels through the utilization of hierarchy
- A successful instantiation of our language control diffusion model, which outperforms the state of the art on the challenging CALVIN benchmark

2. Background

2.1. Reinforcement Learning

We formulate the environment as a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, p)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $\mathcal{R}$, discount factor γ, and transition dynamics p. At each time step t, agents observe a state $s \in \mathcal{S} \subseteq \mathbb{R}^n$, take an action $a \in \mathcal{A} \subseteq \mathbb{R}^m$, and transition to a new state s' with reward r following $s', r \sim p(\cdot, \cdot|s, a)$. The goal of RL is then to learn either a deterministic policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ or a stochastic policy $a \sim \pi(\cdot|s)$ that maximises the policy objective, $J(\pi) = \mathbb{E}_{a\sim\pi, s'\sim p} \sum_{t=0}^{\infty} \gamma^t r_t$.

2.2. Language Conditioned RL

We consider a language-conditioned RL setting where we assume that the true reward function $\mathcal{R}$ is unknown, and must be inferred from a natural language instruction $L \in \mathscr{L}$. Formally, let $\mathcal{F}$ be the function space of $\mathcal{R}$. Then the goal becomes learning an operator from the language instruction to a reward function $\psi : \mathscr{L} \mapsto \mathcal{F}$, and maximizing the policy objective conditioned on the reward function $\psi(L)$: $J(\pi(\cdot \mid s, \mathcal{R})) = \mathbb{E}_{a\sim\pi, s\sim p} \sum_{t=0}^{\infty} \gamma^t r_t$. This formulation can be seen as a contextual MDP (Hallak et al., 2015). Note that the space of tasks that can be specified by language is much larger than that of reward, due to the Markov restriction of the latter (Abel et al., 2021). For example, "pour the milk" and "pour the milk after five o'clock" are both valid instructions, but are indistinguishable from a reward function if the state does not contain temporal information. We assume access to a prior collected dataset $\mathcal{D}$ of N annotated trajectories $\tau_i = \langle(s_0, a_0, ...s_T), L_i\rangle$. The language conditioned policy π_β, or the behavior policy, is defined to be the policy that generates the aforementioned dataset. This general setting can be further restricted to prohibit environment interaction, which recovers offline RL or imitation learning (IL). In this paper, we assume access to a dataset of expert trajectories, such that $\pi_\beta =$ optimal policy $\pi^\star$. Although this may seem like a strong assumption, the setting still poses a challenging learning problem as many unseen states will be encountered during evaluation and must be generalized to. Several prior methods have failed in this setting (De Haan et al., 2019; Ross et al., 2011).

2.3. Goal Conditioned Imitation Learning and Hierarchical RL

Goal-conditioned reinforcement learning (RL) is a subfield of RL that focuses on learning policies that can achieve specific goals or objectives, rather than simply maximizing the cumulative reward. This approach has been extensively studied in various forms in the literature (Arora & Doshi, 2018; Sodhani et al., 2021; Li et al., 2022a; Harrison et al., 2017; Schaul et al., 2015), although we focus on the hierarchical approach (Sutton et al., 1999). Following (Nachum et al., 2018), we formulate the framework in a two-level manner where a high level policy $\pi_{\text{hi}}(\tilde{a}|s)$ samples a goal state $g = \tilde{a}$ every c time steps, which we refer to as the temporal stride. A low level policy (LLP) $\pi_{\text{lo}}(a|s_t, g_t)$ then attempts to reach this state, which can be trained through hindsight relabelling (Andrychowicz et al., 2017) the offline dataset $\mathcal{D}$. In addition, as we consider high dimensional control from pixels, we factorize our low level policy formulation into $\pi_{\text{lo}}(a|s_t, g_t) := \phi(\mathcal{E}(s_t), g_t)$ where $z := \mathcal{E}(s_t)$ defines an encoder function that maps s into a latent representation z and ϕ translates the representation z into a distribution over actions a given a goal g. Note that ϕ is an arbitrary function, and can be made as simple as a matrix or as complex as an another hierarchical model.

3. Language Control Diffusion

3.1. Diffusion Training Objective

We now describe our problem formulation and framework in detail. Since we assume that our dataset is optimal, the policy objective reduces to imitation learning,

$$\min_{\pi} \mathbb{E}_{s,\mathcal{R}\sim\mathcal{D}} \left[D_{\text{KL}} \left(\pi_\beta(\cdot \mid s, \mathcal{R}), \pi(\cdot \mid s, \mathcal{R}) \right) \right]. \tag{1}$$

As we tackle the problem from a planning perspective, we define a state trajectory generator as $\mathcal{P}$ and switch the atomic object from actions to state trajectories $\boldsymbol{\tau} = (s_0, s_1, ..., s_T)$. Thus we aim to minimize the following KL:

$$\begin{aligned} &\min_{\mathcal{P}} D_{\text{KL}} \left(\mathcal{P}_\beta(\boldsymbol{\tau} \mid \mathcal{R}), \mathcal{P}(\boldsymbol{\tau} \mid \mathcal{R}) \right) \\ &= \min_{\mathcal{P}} \mathbb{E}_{\boldsymbol{\tau},\mathcal{R}\sim\mathcal{D}} \left[\log \mathcal{P}_\beta(\boldsymbol{\tau} \mid \mathcal{R}) - \log \mathcal{P}(\boldsymbol{\tau} \mid \mathcal{R}) \right]. \end{aligned} \tag{2}$$

This can be reformulated into the following diffusion training objective:

Algorithm 1 Hierarchical Conditional Diffusion Training

```
Input: baseline goal-conditioned policy $\pi_{\text{lo}} := \phi(\mathcal{E}(s_t), g_t)$, diffusion variance schedule $\alpha_t$, temporal stride $c$, language model $\rho$
Output: trained hierarchical policy $\pi(a_t|s_t) := \pi_{\text{lo}}(a_t|s_t, \pi_{\text{hi}}(g_t|s_t))$, where $g_t$ is sampled every $c$ time steps from $\pi_{\text{hi}}$ as the first state in $\boldsymbol{\tau}^c$.
1: Collect dataset $\mathcal{D}_{\text{onpolicy}}$ by rolling out trajectories $\boldsymbol{\tau} \sim \pi_{\text{lo}}, \rho$
2: Instantiate $\pi_{\text{hi}}$ as diffusion model $\epsilon_\theta(\boldsymbol{\tau}_{\text{noisy}}, t, \rho(L))$
3: repeat
4:    Sample mini-batch $(\boldsymbol{\tau}, L) = B$ from $\mathcal{D}_{\text{onpolicy}}$.
5:    Subsample $\boldsymbol{\tau}^c = (\mathcal{E}(s_0), \mathcal{E}(s_c), \mathcal{E}(s_{2c}), ..., \mathcal{E}(s_T))$.
6:    Sample diffusion step $t \sim \text{Uniform}(\{1, ..., T\})$, noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
7:    Update high level policy $\pi_{\text{hi}}$ with gradient
      $-\nabla_\theta \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{\tau}^c + \sqrt{1-\bar{\alpha}_t}\epsilon, t, \rho(L))\|^2$
8: until converged
```

$$\min_\theta \mathbb{E}_{\boldsymbol{\tau}_0, \boldsymbol{\epsilon}}[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{\tau}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\|^2]. \tag{3}$$

Here t refers to a uniformly sampled diffusion process timestep, $\boldsymbol{\epsilon}_t$ refers to noise sampled from $\mathcal{N}(\mathbf{0}, \mathbf{I})$, $\boldsymbol{\tau}_0$ refers to the original denoised trajectory $\boldsymbol{\tau}$, α_t denotes a diffusion variance schedule and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$. We refer to Appendix B for a more detailed derivation of our objective.

3.2. Near Optimality Guarantees

In order to theoretically justify our approach and show that we can safely do diffusion in the LLP encoder latent space without loss of optimality up to some ϵ, we prove that a near optimal policy is always recoverable with the LLP policy encoder under some mild conditions: that our low-level policy has closely imitated our expert dataset and that the transition dynamics are Lipschitz smooth.

Lipschitz transition dynamics is a mild and typical assumption to make (Asadi et al., 2018; Khalil, 2008). Several continuous control environments will exhibit a Lipschitz transition function, as well as many robotics domains. If an environment is not Lipschitz, it can be argued to be in some sense unsafe (Berkenkamp et al., 2017). Moreover, one could then impose additional action constraints on such an environment to make it Lipschitz again.

Let $\pi_{\text{lo}}(s) := \phi \circ \mathcal{E}(s)$ be a deterministic low-level policy. Similar to Nachum et al. (2019), we can then define the *sub-optimality* of the action and state space induced by $\tilde{\mathcal{A}} = \tilde{\mathcal{S}} = \mathcal{E}(s)$ to be the difference between the best possible high level policy $\pi^*_{\text{hi}} = \text{argmax}_{\pi \in \Pi} J(\pi_{\text{hi}})$ within this new abstract latent space and the best policy in general.

$$\text{SubOpt}(\mathcal{E}, \phi) = \sup_{s \in S} V^{\pi^*}(s) - V^{\pi^*_{\text{hi}}}(s). \tag{4}$$

Intuitively, this captures how much potential performance we stand to lose by abstracting our state and action space.

Proposition 3.1. *If the transition function $p(s'|s, a)$ is Lipschitz continuous with constant K_f and $\sup_{s \in S, a \in A} |\pi_{\text{lo}}(s) - a^*| \leq \epsilon$, then*

$$\text{SubOpt}(\mathcal{E}, \phi) \leq C\epsilon, \tag{5}$$

where $C = \frac{2\gamma}{(1-\gamma)^2} R_{max} K_f \text{dom}(P(s'|s, a))$.

We give a detailed proof in Appendix C.

3.3. Low-Level Policy

We adopt the HULC architecture (Mees et al., 2022a), which is in turn modeled from multi-context imitation learning (MCIL) (Lynch & Sermanet, 2020b). MCIL is a method of training agents to perform a task by imitating multiple experts in different contexts. The goal is to learn a policy that can generalize to unseen situations by training on a diverse set of expert demonstrations. The agent learns to adapt its behavior based on the context in which it is operating, rather than just memorizing the specific actions of the expert in a single context. One of their key insights is leveraging the use of a shared goal space between language and vision through transfer learning methods (Tan et al., 2018). This approach can improve the robustness and flexibility of the trained agent. We utilize this shared latent goal space as the space to perform diffusion within. primarily because it enjoys low dimensionality (32 dimensions) and therefore offers an efficient communication layer between the high level and low level policy.

In HULC, the authors propose an improved version of Multi-Context Imitation Learning (MCIL) by decomposing control into a hierarchical approach. Note that this means that our method is a successful instantiation of a three-level hierarchical model, as we generate the goal states with diffusion *into their high-level policy*. HULC utilizes hierarchy by generating global discrete latent plans and learning local policies that are conditioned on this global plan. Their method also focuses on several key components that have a significant impact on performance, including the architectures used to encode sequences in relabeled imitation learning, the representation of latent distributions, the alignment of language and visual representations, and data augmentation and optimization techniques.

3.4. Model Architecture

It is computationally infeasible to operate directly on the pixel space. Instead, we do planning in a latent state space by utilizing the representation of a goal-conditioned low-level policy. We adopt T5 (Raffel et al., 2020) as our textual encoder, which has similarly found success in the text to image diffusion model Imagen (Ho et al., 2022a). Specifically, we use the T5-XXL variant, which contains 11B parameters and outputs 1024 dimensional embeddings. We utilize a temporal U-Net (Janner et al., 2022), which performs 1D convolutions across the time dimension of the latent plan rather than the 2D convolution typical in text-to-image generation. This is motivated by our desire to preserve equivariance along the time dimension but not the state-action dimension. In addition, we modify the architecture in Janner et al. (2022) by adding conditioning via cross attention in a fashion that resembles the latent diffusion model. When we instantiated the model, we also found that conditioning on the first state through cross attention to be simpler than conditioning through inpainting as is done in the original Diffuser. Finally, we use DDIM (Song et al., 2020a) during inference for increased computational efficiency and faster planning. DDIM uses strided sampling and is able to capture almost the same level of fidelity as typical DDPM sampling (Ho et al., 2020) with an order of magnitude speedup. For rolling out the latent plans generated by the denoiser, we resample a new sequence of goals g with the frequency of c, until either the task is completed successfully or the maximum timestep is reached. Our low-level policy takes over control between samples, with goals generated by the high level policy as input.

4. Experiments

Table 1: Our main result. We compare success rates between our diffusion model and prior benchmarks on multitask long-horizon control (MT-LHC) for 34 disparate tasks. We report the mean and standard deviation across 3 seeds for our method with each seed evaluating for 1000 episodes.

Horizon	GCBC	MCIL	HULC (LLP only)	Ours
1	64.7(4.0)	76.4(1.5)	82.6(2.6)	**88.7**(1.5)
2	28.4(6.2)	48.8(4.1)	64.6(2.7)	**69.9**(2.8)
3	12.2(4.1)	30.1(4.5)	47.9(3.2)	**54.5**(5.0)
4	4.9(2.0)	18.1(3.0)	36.4(2.4)	**42.7**(5.2)
5	1.3(0.9)	9.3(3.5)	26.5(1.9)	**32.2**(5.2)
Avg	1.11(0.3)	1.82(0.2)	2.6(0.1)	**2.88**(0.19)

4.1. Dataset and Metric

In our experiments we aim to answer the following questions: 1) Does a diffusion-based approach perform well for language-conditioned RL? 2) how much efficiency is gained by planning in a latent space? Finally, 3) Is a hierarchical instantiation of the diffusion model really necessary, and what tradeoffs are there to consider? We evaluate on the CALVIN benchmark (Mees et al., 2022b), a challenging multitask long-horizon robotics benchmark. After pretraining our low-level policy on all data, we freeze the policy encoder and train the diffusion U-Net using the frozen encoder. This setup is motivated by the fact that modeling a nonstationary data distribution with the diffusion model is not theoretically sound. We roll out all of our evaluated policies for 1000 trajectories on all 34 tasks, and all comparisons are evaluated in their official repository[1].

4.2. Baselines

As introduced in Section subsection 3.3, we compare against the prior state of the art, HULC and MCIL (Mees et al., 2022a; Lynch & Sermanet, 2020b). To ensure a fair comparison, we rigorously follow their evaluation procedure and build directly from their codebase. MCIL and GCBC results are taken from Mees et al. (2022a), whilst HULC results are reproduced from the original repository.

4.3. Performance of LCD

We outperform prior methods on the challenging multitask long-horizon control (MT-LHC) benchmark, and improve on the strongest prior model's average performance on horizon length one tasks by 6.1% as shown in Table 1. In order to further elucidate why our method works better than HULC, we do a deeper analysis on several failure cases that we observed and show how LCD corrects them. In addition, we significantly outperform HULC in 15 of the 34 tasks, outperform HULC in 23 of the 34 tasks, and improve on the average success rate of single tasks by 3.33% as shown in Table 4. Note that the average success rate differs from Table 1, as the distribution of tasks for MT-LHC is not uniform due to the fact that the CALVIN benchmark filters out infeasible trajectories. A visualization of the first task MT-LHC distribution is provided for future work in Appendix F.

Table 2: Wall clock times for training. Latent dims denotes the size of the latent space that we perform the diffusion generation in. We compare against two variants of Diffuser. Diffuser-1D is the same model as presented in Table 1 which utilizes a VAE trained from scratch on the dataset, whilst Diffuser-2D utilizes a large pretrained VAE from Stable Diffusion (Rombach et al., 2022). Inference time (sec) refers to the average amount of time taken in seconds to produce an action.

	Ours (HLP only)	HULC	Ours (full)	Diffuser-1D	Diffuser-2D
Training (hrs)	13.3	82	95.3	20.8	49.2
Inference time (sec)	0.333	0.005	0.336	1.11	5.02
Model size	20.1M	47.1M	67.8M	74.7M	125.5M
Latent dims	32	N/A	32	512	1024
Avg ∇ updates/sec	6.25	.5	6.25	4	2.1

4.4. Task Generalization

We test the task generalization of LCD on a collection of five held out tasks in Table 3. Note that the success rates given

[1]https://github.com/lukashermann/hulc/tree/fb14d5461ae54f919d52c0c30131b38f806ef8db

here differ from Table 4, as the initial state distribution differ here. The initial states here come directly from CALVIN's initial state generator, whereas the initial states in Table 4 may be from the last state of a prior task.

Table 3: Task generalization of LCD on a collection of five held out tasks. Note that the success rates given here differ from Table 4, as the initial state distribution also varies. The initial states here come directly from CALVIN's initial state generator, whereas the initial states in Table 4 may arise from the last state of a prior task.

Task	Ours
Lift Pink Block Table	55.00(16.33)
Lift Red Block Slider	88.33(8.50)
Push Red Block Left	35.00(7.07)
Push Into Drawer	90.00(10.80)
Rotate Blue Block Right	36.67(14.34)
Avg SR	61.00(7.79)

4.5. Efficiency

Through the usage of DDIM, temporal abstraction, and low-dimensional generation, we find in Table 2 that LCD is significantly faster during rollout and training than Diffuser. In addition, our method is significantly faster to train than HULC, albeit slower during rollout. All numbers are taken from our own experiments and server for reproducibility and fairness, including baselines. Baselines are run with 8 2080 Ti GPUs following the original author guidelines, whilst our experiments are run with a single 2080 Ti.

4.6. Is diffusion really necessary?

In order to analyze whether diffusion actually improves HULC or if the gain comes just from the usage of a high-level policy, we perform an ablation study in Table 5 by using a simple MLP as a high-level policy, which receives T5-language embeddings as well as the current state, and attempts to predict the next goal state. We find that this ablation significantly underperforms LCD, and slightly underperforms HULC.

4.7. Robustness to Hyperparameters

In Figure 2 we show that our diffusion method is robust to several hyperparameters including the number of diffusion steps, a frame offset o, and the hidden dimensions of the model. The number of diffusion steps is typically a key parameter determining the quality of the representations. However, we found that the performance of our method is relatively insensitive to the number of function evaluations. Additionally, we consider augmenting our dataset by adding a frame offset o during the sampling of our goal state s_{c+o} for potentially improving generalization. However, we find that this effect is more or less negligible. Finally, our method is also robust with respect to the number of parameters, and

Table 4: Comparison of success rates (SR) across single tasks, evaluated with HULC and LCD. Statistically significant differences at $P < .05$ in SR are bolded. Testing is done with a two-sample independent t-test. We again report the performance for the mean and standard deviation across 3 seeds for all methods, with runs taken from the MT-LHC benchmark in Table 1.

Task	HULC	Ours
Close Drawer	99.50(0.36)	95.97(3.46)
Lift Blue Block Drawer	86.97(13.01)	90.83(9.47)
Lift Blue Block Slider	58.80(6.99)	**78.77(5.20)**
Lift Blue Block Table	72.73(7.89)	76.07(2.03)
Lift Pink Block Drawer	67.93(9.39)	76.27(22.80)
Lift Pink Block Slider	71.50(4.54)	**79.30(2.01)**
Lift Pink Block Table	68.00(5.60)	70.47(3.07)
Lift Red Block Drawer	**91.67(11.79)**	77.00(4.71)
Lift Red Block Slider	71.97(2.57)	79.70(7.32)
Lift Red Block Table	60.87(5.84)	**69.43(2.77)**
Move Slider Left	96.27(1.14)	96.27(1.96)
Move Slider Right	99.27(0.24)	97.67(2.19)
Open Drawer	97.83(1.43)	98.07(0.50)
Place In Drawer	95.93(1.37)	95.90(1.79)
Place In Slider	70.90(5.44)	**81.10(0.14)**
Push Blue Block Left	60.10(9.69)	71.93(6.82)
Push Blue Block Right	27.60(9.10)	33.10(2.86)
Push Into Drawer	72.13(5.74)	75.47(5.29)
Push Pink Block Left	66.73(17.27)	64.80(11.98)
Push Pink Block Right	57.67(4.46)	51.17(6.04)
Push Red Block Left	64.17(4.71)	55.30(16.32)
Push Red Block Right	28.73(16.07)	37.83(3.27)
Rotate Blue Block Left	59.63(7.38)	**70.03(2.56)**
Rotate Blue Block Right	62.57(4.17)	**72.77(6.76)**
Rotate Pink Block Left	71.53(6.99)	75.07(9.15)
Rotate Pink Block Right	55.93(4.29)	**64.00(4.84)**
Rotate Red Block Left	67.43(5.15)	**81.23(8.86)**
Rotate Red Block Right	69.53(5.44)	74.57(2.11)
Stack Block	**36.83(2.84)**	32.00(3.56)
Turn Off Led	98.30(1.87)	96.87(1.80)
Turn Off Lightbulb	98.33(1.25)	95.67(3.78)
Turn On Led	98.73(0.93)	97.63(1.73)
Turn On Lightbulb	**98.97(0.40)**	95.60(1.58)
Unstack Block	94.53(4.53)	90.80(3.72)
Unstack Block	94.53(4.53)	90.80(3.72)
Avg Success Rate	70.68(2.65)	74.01(2.64)

instantiating a larger model does not induce overfitting.

4.8. Summary of critical findings

Diffuser fails to plan in original state space

After investigating the performance of diffuser, our findings in Figure **??** indicate that Diffuser fails to successfully replan in high dimensional state spaces when using a Variational Autoencoder (VAE). Based on our observations, we hypothesize that the failure of Diffuser to replan successfully is due to the diffusion objective enabling interpolation between low density regions in the VAE's latent space by the nature of the training objective smoothing the original probability distribution of trajectories. In practice, this interpolation between low density regions corresponds to physically infeasible trajectories. This suggests that interpolation between low density regions in the VAE's latent space is a significant factor in the failure of diffuser to plan and

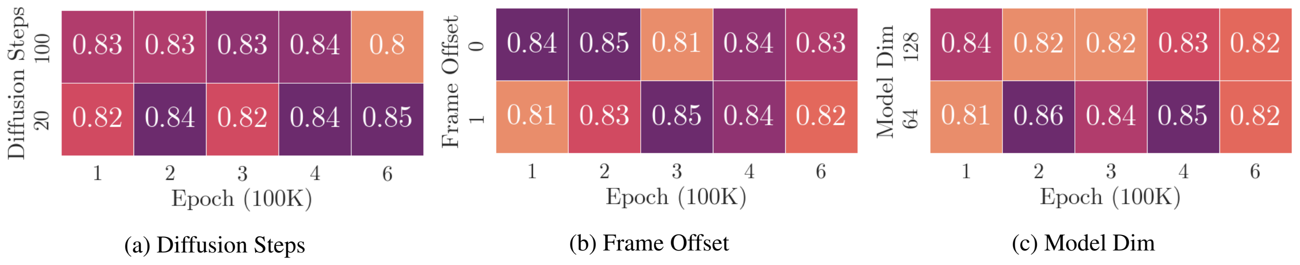


Figure 2: Single task success rates from MT-LHC across different epochs and hyperparameters. Our diffusion model is robust to various hyperparameters such as diffusion steps, frame offset, hidden dimensions, and number of parameters. While the number of diffusion steps usually affects the quality of representations, our method is relatively insensitive to it. Adding a frame offset during sampling to augment the dataset has little effect on generalization. Furthermore, a larger model does not lead to overfitting.

Table 5: Ablation of our method by comparing against a simple three layer MLP as high level policy. We use the same methodology as in Table 1, and report the mean and standard deviation across 3 seeds for our method with each seed evaluating for 1000 episodes. Even given the ground truth validation language embeddings, the MLP underperforms the original model and LCD.

Horizon Length	MLP	Ours
1	82.1(4.0)	**88.7**(1.5)
2	61.0(6.2)	**69.9**(2.8)
3	49.2(4.1)	**54.5**(5.0)
4	32.1(2.0)	**42.7**(5.2)
5	25.6(0.9)	**32.2**(5.2)
Avg. length	2.59(0.01)	**2.81**(0.12)

replan successfully. Our results highlight the need for better representation, and for further research of scaling diffusion models to high dimensional control.

Representation is essential for effective diffusion

We find that having a good representation is essential for effective diffusion in general control settings. A good representation greatly eases the learning difficulty on the diffusion model by reducing the number of dimensions needed to model, which in turn increases the information density and makes it easier for the model to learn the underlying dynamics of the system. The good representation likely massages the regions of low density latent space between feasible trajectories into areas that still correspond to physically feasible regions when decoded by the low level policy. This enables the model to generalize well to unseen situations, improving its robustness and flexibility. Additionally, by having a good representation, the diffusion model is able to successfully generalize to new scenarios, which is crucial for the model's effectiveness in a real-world application. Our results highlight the importance of latent representation in the diffusion process.

The encoder representation of deep goal-conditioned RL models can be effectively used for hierarchical RL

Our results imply that the encoder representation of deep reinforcement learning models have the potential to be effectively utilized in hierarchical reinforcement learning. In modern deep policy learning from pixels, we propose an effective task-aware representation can be extracted by using the latent space of an intermediate layer as an encoder. Although we have demonstrated the effectiveness of this approach by instantiating it successfully in a hierarchical diffusion model, this approach is general and can be applied to any generative model in hierarchical RL. our results suggest that using a shared policy encoder between the high and low level policies can improve the effectiveness and efficiency of generative modelling in hierarchical RL.

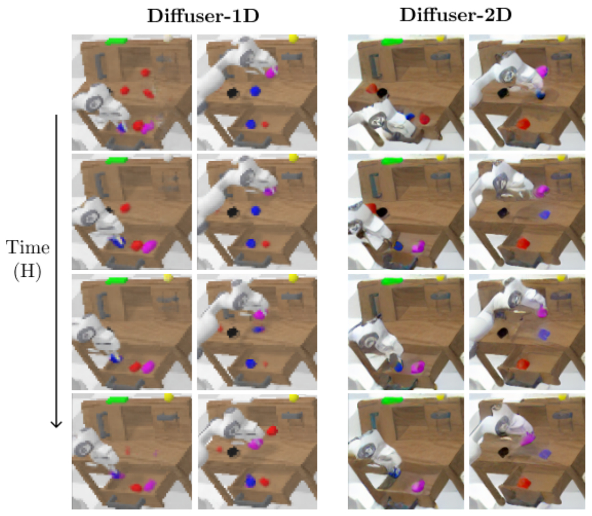


Figure 3: Denoised Latent Representations. Directly using latent diffusion models fails. Hallucination occurs on a β-TC VAE trained from scratch on the CALVIN dataset (Diffuser-1D), and loss of fine details occurs with SD v1.4's (Rombach et al., 2022) internet-scale pretrained autoencoder (Diffuser-2D). For more and enlarged samples please refer to Appendix G.

5. Related Work

Text-to-Control Models Text-to-control models or language-conditioned policies (Lynch & Sermanet, 2020a; Jiang et al., 2019; Colas et al., 2020) have been explored in the RL community for improving generalization to novel tasks and environments (Hermann et al., 2017; Bahdanau et al., 2018; Hill et al., 2019; Colas et al., 2020; Wang et al., 2020a). Although they are not the only way to incorporate external knowledge in the form of text to decision making tasks (Luketina et al., 2019; Zhong et al., 2019), they remain one of the most popular as they are simple to evaluate and enjoy great interest from several jointly related fields including Vision Language Navigation (VLN) (Wang et al., 2020b; Zheng et al., 2022) and Embodied AI (Duan et al., 2022). However, language grounding in RL remains notoriously difficult, as language vastly underspecifies all possible configurations of a corresponding state (Quine, 1960). Modern text to control models often still struggle with long-horizon language commands and misinterpret the language instruction. Hu et al. (2019) attempt to solve a long-horizon Real-Time Strategy (RTS) game with a hierarchical method utilizing language as the communication interface between the high level and low level policy, whilst Harrison et al. (2017) consider training a language encoder-decoder for policy shaping, Hanjie et al. (2021) utilize an attention module conditioned on textual entities for strong generalization and better language grounding. Zhong et al. (2022) propose a model-based objective to deal with sparse reward settings with language descriptions and Mu et al. (2022) also tackle the sparse reward settings through the usage of intrinsic motivation with bonuses added on novel language descriptions. However, only Hu et al. (2019) consider the long-horizon setting, whom do not consider a high-dimensional state space. Jang et al. (2022) carefully examine the importance of diverse and massive data collection in enabling task generalization through language and propose a FiLM conditioned CNN-MLP model (Perez et al., 2018). Much work has applied using more data and compute for control (Brohan et al., 2022; Jaegle et al., 2021; Reed et al., 2022). However, none of these methods consider the usage of diffusion models as the medium between language and RL.

Diffusion Models Diffusion models such as DALL-E 2 (Ramesh et al., 2022) and GLIDE (Nichol et al., 2022) have recently shown promise as generative models, with state-of-the-art text-to-image generation results demonstrating a surprisingly deep understanding of semantic relationships between objects and the high fidelity generation of novel scenes. Stable diffusion, an instantiation of latent diffusion (Rombach et al., 2022), has also achieved great success and is somewhat related to our method as they also consider performing the denoising generation in a smaller latent space, albeit with a variational autoencoder (Kingma & Welling, 2013) rather than a low level policy encoder.

One driver of this recent success in generative modeling is the usage of *classifier-free guidance*, which is amenable to the RL framework through the usage of language as a reward function. The language command is a condition for optimal action generation, which draws direct connection to control as inference (Levine, 2018).

Given the success of denoising diffusion probabilistic models (Ho et al., 2020) in text-to-image synthesis (Sohl-Dickstein et al., 2015a), the diffusion model has been further explored in both discrete and continuous data domains, including image and video synthesis (Ho et al., 2022b;c), text generation (Li et al., 2022b), and time series (Rasul et al., 2021). Video generation are especially relevant to this work, as they are a direct analogue of diffusion planning model Diffuser (Janner et al., 2022) in pixel space without actions.

Diffuser first proposed to transform decision making into inpainting and utilize diffusion models to solve this problem, which much work has followed up on (Dai et al., 2023; Chi et al., 2023; Ajay et al., 2022). Specifically, they diffuse the state and actions jointly for imitation learning and goal-conditioned reinforcement learning through constraints specified through classifier guidance, and utilize Diffuser for solving long-horizon and task compositional problems in planning. Instead of predicting the whole trajectory for each state, (Wang et al., 2022b) apply the diffusion model to sample a single action at a time conditioned by state. However, neither of these works consider control from pixels, or utilizing language for generalization.

6. Conclusion

Learning atomic sub-skills through language is critical to scaling to more complex and open environments. We explore solving this problem through learned state and temporal abstractions, and show that the strengths of diffusion models can be leveraged for long horizon plans, and their weakness at low-level detail generation can be managed through learning low-level, goal-conditioned policies with imitation learning. Experiments and qualitative analysis demonstrate the simplicity and effectiveness of our model, showing that LCD is able to achieve state-of-the-art performance on a competitive language-conditioned control benchmark from rich observations, over long-horizons, and generalize to new scenarios.

For future work, one could extend LCD to incorporate additional levels of hierarchy and scale to even longer horizons. Additionally, a deeper analysis on the interplay of different representations and the diffusion model could be performed. Finally, one could further probe the task generalization, and potentially improve generalization on the language side by leveraging better pre-trained models, and incorporating that distilled knowledge to improve reasoning and planning for control.

Acknowledgements

We would like to thank Michael Janner, Yilun Du, and Weixi Feng for fruitful discussions, Oier Mees and Luka Shermann for clarifications and support on the CALVIN benchmark, and Ashley Jiang for inspiring and creating some early drafts of the main figures.

References

Abel, D., Dabney, W., Harutyunyan, A., Ho, M. K., Littman, M., Precup, D., and Singh, S. On the Expressivity of Markov Reward. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. S., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 7799–7812. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/4079016d940210b4ae9ae7d41c4a2065-Paper.pdf.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Ho, D., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jang, E., Ruano, R. J., Jeffrey, K., Jesmonth, S., Joshi, N. J., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.-H., Levine, S., Lu, Y., Luu, L., Parada, C., Pastor, P., Quiambao, J., Rao, K., Rettinghouse, J., Reyes, D., Sermanet, P., Sievers, N., Tan, C., Toshev, A., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Xu, S., Yan, M., and Zeng, A. Do as i can, not as i say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

Ajay, A., Du, Y., Gupta, A., Tenenbaum, J., Jaakkola, T., and Agrawal, P. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

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

Arora, S. and Doshi, P. A survey of inverse reinforcement learning: Challenges, methods and progress, 2018. URL https://arxiv.org/abs/1806.06877.

Asadi, K., Misra, D., and Littman, M. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 264–273. PMLR, 2018.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Hosseini, A., Kohli, P., and Grefenstette, E. Learning to understand goal specifications by modelling reward. *arXiv preprint arXiv:1806.01946*, 2018.

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems*, 30, 2017.

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

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

Colas, C., Akakzia, A., Oudeyer, P.-Y., Chetouani, M., and Sigaud, O. Language-conditioned goal generation: a new approach to language grounding for rl. *LAREL Workshop 2020*, abs/2006.07043, 2020.

Dai, Y., Yang, M., Dai, B., Dai, H., Nachum, O., Tenenbaum, J., Schuurmans, D., and Abbeel, P. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

De Haan, P., Jayaraman, D., and Levine, S. Causal confusion in imitation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Duan, J., Yu, S., Tan, H. L., Zhu, H., and Tan, C. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.

Hallak, A., Di Castro, D., and Mannor, S. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Hanjie, A. W., Zhong, V. Y., and Narasimhan, K. Grounding language to entities and dynamics for generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 4051–4062. PMLR, 2021.

Harrison, B., Ehsan, U., and Riedl, M. O. Guiding reinforcement learning exploration using natural language. *arXiv preprint arXiv:1707.08616*, 2017.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., et al. Grounded language learning in a simulated 3D world. *arXiv preprint arXiv:1706.06551*, 2017.

Hill, F., Lampinen, A., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A. Emergent systematic generalization in a situated agent, 2019.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J., et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022a.

Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., and Salimans, T. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022b. URL `http://jmlr.org/papers/v23/21-0635.html`.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models, 2022c. URL `https://arxiv.org/abs/2204.03458`.

Hu, H., Yarats, D., Gong, Q., Tian, Y., and Lewis, M. Hierarchical decision making by generating and following natural language instructions. *Advances in neural information processing systems*, 32, 2019.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *ICML*, 2022a.

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

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

Jang, E., Irpan, A., Khansari, M., Kappler, D., Ebert, F., Lynch, C., Levine, S., and Finn, C. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Janner, M., Du, Y., Tenenbaum, J. B., and Levine, S. Planning with diffusion for flexible behavior synthesis, 2022. URL `https://arxiv.org/abs/2205.09991`.

Jiang, Y., Gu, S., Murphy, K., and Finn, C. *Language as an Abstraction for Hierarchical Deep Reinforcement Learning*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Jiang, Y., Gupta, A., Zhang, Z., Wang, G., Dou, Y., Chen, Y., Fei-Fei, L., Anandkumar, A., Zhu, Y., and Fan, L. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2022.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Khalil, H. K. Nonlinear systems third edition. 2008.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Li, S., Puig, X., Du, Y., Wang, C., Akyurek, E., Torralba, A., Andreas, J., and Mordatch, I. Pre-trained language models for interactive decision-making. *arXiv preprint arXiv:2202.01771*, 2022a.

Li, X. L., Thickstun, J., Gulrajani, I., Liang, P., and Hashimoto, T. B. Diffusion-lm improves controllable text generation, 2022b. URL `https://arxiv.org/abs/2205.14217`.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., Whiteson, S., and Rocktäschel, T. A survey of reinforcement learning informed by natural language. *arXiv preprint arXiv:1906.03926*, 2019.

Lynch, C. and Sermanet, P. Grounding language in play. *CoRR*, abs/2005.07648, 2020a. URL `https://arxiv.org/abs/2005.07648`.

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

Mees, O., Hermann, L., and Burgard, W. What matters in language conditioned robotic imitation learning over unstructured data, 2022a. URL `https://arxiv.org/abs/2204.06252`.

Mees, O., Hermann, L., Rosete-Beas, E., and Burgard, W. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 2022b.

Mu, J., Zhong, V., Raileanu, R., Jiang, M., Goodman, N., Rocktäschel, T., and Grefenstette, E. Improving intrinsic exploration with language abstractions. *arXiv preprint arXiv:2202.08938*, 2022.

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

Nachum, O., Gu, S., Lee, H., and Levine, S. Near-optimal representation learning for hierarchical reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=H1emus0qF7.

Nichol, A. Q., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., Mcgrew, B., Sutskever, I., and Chen, M. GLIDE: Towards photorealistic image generation and editing with text-guided diffusion models. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16784–16804. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/nichol22a.html.

Perez, E., Strub, F., De Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Quine, W. V. O. Word and object mit press. *Cambridge MA*, 1960.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

Rasul, K., Seward, C., Schuster, I., and Vollgraf, R. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. 2021. doi: 10.48550/ARXIV.2101.12072. URL https://arxiv.org/abs/2101.12072.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

Singer, U., Polyak, A., Hayes, T., Yin, X., An, J., Zhang, S., Hu, Q., Yang, H., Ashual, O., Gafni, O., et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

Sodhani, S., Zhang, A., and Pineau, J. Multi-task reinforcement learning with context-based representations, 2021. URL https://arxiv.org/abs/2102.06177.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015a. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015b.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

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

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

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, pp. 270–279. Springer, 2018.

Wang, H., Wu, Q., and Shen, C. Soft expert reward learning for vision-and-language navigation. In *European Conference on Computer Vision*, pp. 126–141. Springer, 2020a.

Wang, T., Du, S. S., Torralba, A., Isola, P., Zhang, A., and Tian, Y. Denoised mdps: Learning world models better than the world itself. *arXiv preprint arXiv:2206.15477*, 2022a.

Wang, X., Huang, Q., Celikyilmaz, A., Gao, J., Shen, D., Wang, Y.-F., Wang, W. Y., and Zhang, L. Vision-language navigation policy learning and adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 43 (12):4205–4216, 2020b.

Wang, Z., Hunt, J. J., and Zhou, M. Diffusion policies as an expressive policy class for offline reinforcement learning, 2022b. URL `https://arxiv.org/abs/2208.06193`.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics.

Weng, L. What are diffusion models? *lilianweng.github.io*, Jul 2021. URL `https://lilianweng.github.io/posts/2021-07-11-diffusion-models/`.

Zhang, A., McAllister, R., Calandra, R., Gal, Y., and Levine, S. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

Zheng, K., Chen, X., Jenkins, O. C., and Wang, X. E. Vlmbench: A compositional benchmark for vision-and-language manipulation. *arXiv preprint arXiv:2206.08522*, 2022.

Zhong, V., Rocktäschel, T., and Grefenstette, E. Rtfm: Generalising to novel environment dynamics via reading. *arXiv preprint arXiv:1910.08210*, 2019.

Zhong, V., Mu, J., Zettlemoyer, L., Grefenstette, E., and Rocktäschel, T. Improving policy learning via language dynamics distillation. *arXiv preprint arXiv:2210.00066*, 2022.

Appendix

Outline of the Appendix

- Appendix A presents experiment details, HP settings and corresponding experiment results.
- Appendix B presents the derivation for the trajectory based denoising training objective.
- Appendix C presents the proof for Theorem 3.1.
- Appendix D presents the details for LCD's inference procedure.
- Appendix E presents an example figure of the denoising process for the Diffuser-2D model.
- Appendix F presents the task distributions of the MT-LHC benchmark results given in Table 1.
- Appendix G offers more samples with higher resolution of the representation failures of Diffuser-1D and Diffuser-2D, first referenced in Figure **??**.
- Appendix H presents a comparison between the TSNE visualizations of a ground-truth encoded trajectory and one gneerated from Diffuser-1D.
- Appendix I presents a TSNE visualization of the discrete latent plan space within HULC. We clarify that this is not the latent goal space that our model does generation in.

Please refer to our website `https://lcd.eddie.win` for more qualitative results in video format. We release our code implementation at `https://github.com/ezhang7423/language-control-diffusion`.

A. Hyper-parameter settings and training details

For all methods we proposed in Table 1, Table 3, Table 4, and Table 6, we obtain the mean and standard deviation of each method across 3 seeds. Each seed contains the individual training process and evaluates the policy for 1000 episodes.

A.1. HP and training details for methods in Table 1 and Table 4.

Model	Module	Hyperparameter	Value
HULC	Trainer	Max Epochs	30
		β for KL Loss	0.01
		λ for Contrastive Loss	3
		Optimizer	Adam
		Learning Rate	2e-4
	Model	Transformer Hidden Size	2048
		Language Embedding Size	384
LCD	Gaussian Diffusion	Action Dimension	32
		Action Weight	10
		Loss Type	L2
		Observation Dimension	32
		Diffusion Steps	20
		Model Dimension	64
	Trainer	EMA Decay	0.995
		Label Frequency	200000
		Sample Frequency	1000
		Batch Size	512
		Learning Rate	2e-4
		Train Steps	250k
		Number of Steps Per Epoch	10000
		Normalizer	Gaussian Normalizer
		Frame Offset	0

Table 6: Hyperparameters for our methods in Table 1 and Table 4.

B. Training Objective Derivation

To model this, we turn to diffusion models (Weng, 2021), whom we borrow much of the following derivation from. Inspired by non-equilibrium thermodynamics, the common forms of diffusion models (Sohl-Dickstein et al., 2015b; Ho et al., 2020; Song et al., 2020a) propose modeling the data distribution $p(\boldsymbol{\tau})$ as a random process that steadily adds increasingly varied amounts of Gaussian noise to samples from $p(\boldsymbol{\tau})$ until the distribution converges to the standard normal. We denote the forward process as $f(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{t-1})$, with a sequence of variances $(\beta_0, \beta_1 ... \beta_T)$. We define $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$.

$$f(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0) = \prod_{t=1}^{T} f(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{t-1}), \quad \text{where } f(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{t-1}) = \mathcal{N}(\boldsymbol{\tau}_t; \sqrt{1-\beta_t}\boldsymbol{\tau}_{t-1}, \beta_t \mathbf{I}). \tag{6}$$

One can tractably reverse this process when conditioned on $\tau_{\mathbf{0}}$, which allows for the construction of a sum of the typical variational lower bounds for learning the backward process' density function (Sohl-Dickstein et al., 2015b). Since the backwards density also follows a Gaussian, it suffices to predict μ_θ and Σ_θ which parameterize the backwards distribution:

$$p_\theta\left(\boldsymbol{\tau}_{t-1} \mid \boldsymbol{\tau}_t\right) = \mathcal{N}\left(\boldsymbol{\tau}_{t-1}; \boldsymbol{\mu}_\theta\left(\boldsymbol{\tau}_t, t\right), \boldsymbol{\Sigma}_\theta\left(\boldsymbol{\tau}_t, t\right)\right). \tag{7}$$

In practice, Σ_θ is often fixed to constants, but can also be learned through reparameterization. Following (Ho et al., 2020) we consider learning only μ_θ, which can be computed just as a function of τ_t and $\epsilon_\theta(\tau_t, t)$. One can derive that $\boldsymbol{\tau}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{\tau}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ for $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, through a successive reparameterization of (6) until arriving at $f(\boldsymbol{\tau}_t|\boldsymbol{\tau}_0)$. Therefore to sample from $p(\boldsymbol{\tau})$, we need only to learn ϵ_θ, which is done by regressing to the ground truth ϵ given by the tractable backwards density. Assuming we have ϵ_θ, we can then follow a Markov chain of updates that eventually converges to the original data distribution, in a procedure reminiscent of Stochastic Gradient Langevin Dynamics (Welling & Teh):

$$\boldsymbol{\tau}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{\tau}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{\tau}_t, t\right)\right) + \sigma_t \mathbf{z}, \quad \text{where } \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{8}$$

To learn ϵ_θ, we can minimize the following variational lower bound on the negative log-likelihood:

$$\begin{aligned}
L_{\text{CE}} &= -\mathbb{E}_{q(\boldsymbol{\tau}_0)} \log p_\theta(\boldsymbol{\tau}_0) \\
&= -\mathbb{E}_{q(\boldsymbol{\tau}_0)} \log \Big(\int p_\theta(\boldsymbol{\tau}_{0:T}) d\boldsymbol{\tau}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\boldsymbol{\tau}_0)} \log \Big(\int q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0) \frac{p_\theta(\boldsymbol{\tau}_{0:T})}{q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0)} d\boldsymbol{\tau}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\boldsymbol{\tau}_0)} \log \Big(\mathbb{E}_{q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0)} \frac{p_\theta(\boldsymbol{\tau}_{0:T})}{q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0)} \Big) \\
&\leq -\mathbb{E}_{q(\boldsymbol{\tau}_{0:T})} \log \frac{p_\theta(\boldsymbol{\tau}_{0:T})}{q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0)} \\
&= \mathbb{E}_{q(\boldsymbol{\tau}_{0:T})} \Big[\log \frac{q(\boldsymbol{\tau}_{1:T}|\boldsymbol{\tau}_0)}{p_\theta(\boldsymbol{\tau}_{0:T})} \Big] = L_{\text{VLB}}. \\
L_{\text{VLB}} &= L_T + L_{T-1} + \cdots + L_0 \\
\text{where } L_T &= D_{\text{KL}}(q(\boldsymbol{\tau}_T|\boldsymbol{\tau}_0) \parallel p_\theta(\boldsymbol{\tau}_T)) \\
L_t &= D_{\text{KL}}(q(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{t+1}, \boldsymbol{\tau}_0) \parallel p_\theta(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{t+1})) \\
&\qquad \text{for } 1 \leq t \leq T-1 \text{ and} \\
L_0 &= -\log p_\theta(\boldsymbol{\tau}_0|\boldsymbol{\tau}_1).
\end{aligned} \tag{9}$$

Which enables us to find a tractable parameterization for training, as the KL between two Gaussians is analytically computable.

$$
\begin{aligned}
L_t &= \mathbb{E}_{\boldsymbol{\tau}_0,\boldsymbol{\epsilon}}\Big[\frac{1}{2\|\boldsymbol{\Sigma}_\theta(\boldsymbol{\tau}_t,t)\|_2^2}\|\tilde{\boldsymbol{\mu}}_t(\boldsymbol{\tau}_t,\boldsymbol{\tau}_0)-\boldsymbol{\mu}_\theta(\boldsymbol{\tau}_t,t)\|^2\Big] \\
&= \mathbb{E}_{\boldsymbol{\tau}_0,\boldsymbol{\epsilon}}\Big[\frac{1}{2\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\frac{1}{\sqrt{\alpha_t}}\Big(\boldsymbol{\tau}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\Big)-\frac{1}{\sqrt{\alpha_t}}\Big(\boldsymbol{\tau}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}_t,t)\Big)\|^2\Big] \\
&= \mathbb{E}_{\boldsymbol{\tau}_0,\boldsymbol{\epsilon}}\Big[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\boldsymbol{\epsilon}_t-\boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}_t,t)\|^2\Big] \\
&= \mathbb{E}_{\boldsymbol{\tau}_0,\boldsymbol{\epsilon}}\Big[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\boldsymbol{\epsilon}_t-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{\tau}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t,t)\|^2\Big].
\end{aligned}
\tag{10}
$$

After removing the coefficient at the beginning of this objective following (Ho et al., 2020), we arrive at the objective used in the practical algorithm 1:

$$
\mathbb{E}_{\boldsymbol{\tau}_0,\boldsymbol{\epsilon}}[\|\boldsymbol{\epsilon}_t-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{\tau}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t,t)\|^2]. \tag{11}
$$

Furthermore, thanks to the connection between noise conditioned score networks and diffusion models (Song et al., 2020b; Ho et al., 2020), we are able to state that $\epsilon_\theta \propto -\nabla \log p(\boldsymbol{\tau})$:

$$
\begin{aligned}
\mathbf{s}_\theta(\boldsymbol{\tau}_t,t) &\approx \nabla_{\boldsymbol{\tau}_t}\log p(\boldsymbol{\tau}_t) \\
&= \mathbb{E}_{q(\boldsymbol{\tau}_0)}[\nabla_{\boldsymbol{\tau}_t}p(\boldsymbol{\tau}_t|\boldsymbol{\tau}_0)] \\
&= \mathbb{E}_{q(\boldsymbol{\tau}_0)}\Big[-\frac{\boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}_t,t)}{\sqrt{1-\bar{\alpha}_t}}\Big] \\
&= -\frac{\boldsymbol{\epsilon}_\theta(\boldsymbol{\tau}_t,t)}{\sqrt{1-\bar{\alpha}_t}}.
\end{aligned}
\tag{12}
$$

Therefore, by using a variant of ϵ_θ conditioned on language to denoise our latent plans, we can effectively model $-\nabla_\tau \mathcal{P}_\beta(\boldsymbol{\tau} \mid \mathcal{R})$ with our diffusion model, iteratively guiding our generated trajectory towards the optimal trajectories conditioned on language.

C. Proof of 3.1

Proof. The proof is fairly straightforward, and can be shown by translating our definition of suboptimality into the framework utilized by (Nachum et al., 2019). We are then able to leverage their first theorem bounding suboptimality by the Total Variation (TV) between transition distributions to show our result, as TV is bounded by the Lipschitz constant multiplied by the domain of the function.

(Nachum et al., 2019) first define a low level policy generator Ψ which maps from $S \times \tilde{A}$ to Π. Using the high level policy to sample a goal $g_t \sim \pi_{\mathrm{hi}}(g|s_t)$, they use Ψ to translate this to a policy $\pi_t = \Psi(s_t, g_t)$, which samples actions $a_{t+k} \sim \pi_t(a|s_{t+k}, k)$ from $k \in [0, c-1]$. The process is repeated from s_{t+c}. Furthermore, they define an inverse goal generator $\varphi(s, a)$, which infers the goal g that would cause Ψ to yield an action $\tilde{a} = \Psi(s, g)$. The following can then be shown:

Theorem C.1. *If there exists $\varphi : S \times A \to \tilde{A}$ such that,*

$$\sup_{s \in S, a \in A} D_{TV}(P(s'|s,a) || P(s'|s, \Psi(s, \varphi(s,a)))) \leq \epsilon, \tag{13}$$

then $\mathrm{SubOpt}'(\Psi) \leq C\epsilon$, *where* $C = \frac{2\gamma}{(1-\gamma)^2} R_{max}$.

Note that their SubOpt' is different from ours; whilst we defined in terms of the encoder $\mathcal{E}$ and action generator ϕ, they define it in terms of Ψ. Note, however, that the two are equivalent when the temporal stride $c = 1$, as Ψ becomes $\pi_{\mathrm{lo}} = \phi \circ \mathcal{E}$. It is essential to note that when using a goal conditioned imitation learning objective, as we do in this paper, π_{lo} becomes equivalent to an inverse dynamics model IDM$(s, \mathcal{E}(s)) = a$ and that $\varphi(s, a)$ becomes equivalent to $\mathcal{E}(s')$. This is the key to our proof, as the second term in the total variation of C.1 reduces to

$$\begin{aligned} &P(s'|s, \Psi(s, \varphi(s,a)))) \\ &= P(s'|s, \Psi(s, \mathcal{E}(s'))) \\ &= P(s'|s, a + \epsilon)). \end{aligned} \tag{14}$$

Since we have that the transition dynamics are Lipschitz:

$$\begin{aligned} &\int_{\mathcal{A},\mathcal{S}} |P(s'|s,a) - P(s'|s, a+\epsilon))| \, d\nu \\ &\leq \int_{\mathcal{A},\mathcal{S}} K_f |a - (a+\epsilon)| \, d\nu \\ &= K_f \epsilon \int_{\mathcal{A},\mathcal{S}} d\nu \\ &= K_f \epsilon \, \mathrm{dom}(P(s'|s,a)) \end{aligned} \tag{15}$$

Which we can then plug into 13 to obtain the desired $C = \frac{2\gamma}{(1-\gamma)^2} R_{max} K_f \mathrm{dom}(P(s'|s,a))$. □

D. Inference Pipeline

We give an overview of LCD's inference pipeline below. For our ablations, we are also able to use this pipeline to decode the latent states into pixel space to analyze and interpret the plans generated by the denoising autoencoder. We give examples of these decoded plans in Appendix G.

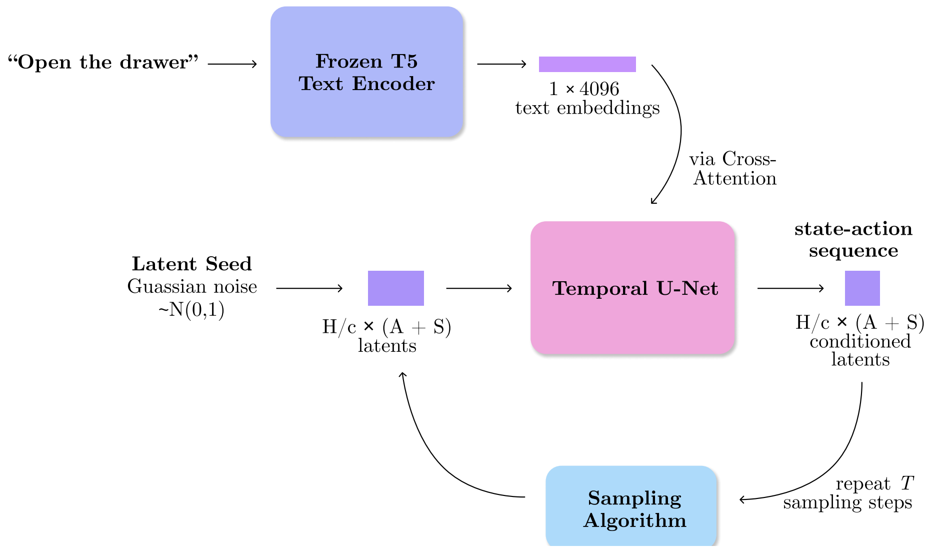


Figure 4: An overview of our inference pipeline. We pass sampled noise to our denoising autoencoder along with an encoded language prompt. The autoencoder is instantiated as a temporal U-Net. By repeating this process iteratively, we are able to generate high-fidelity latent plans conditioned on language.

E. Diffuser-2D

Here we give details for our strongest Diffusion-based ablation, which uses Stable Diffusion's VAE for generating latent plans, which outputs a latent 2D feature map, with height and width 1/8 of the original image. Plans are sampled with a temporal stride of 7, such that each trajectory covers a total of 63 timesteps with $t = 0, 7, 14 \ldots 63$. Overall, generation quality tends to be higher and more temporally coherent than that of the 1D model, but low level details still not precise enough for control from pixels. For examples of model outputs, please refer to subsection G.2.

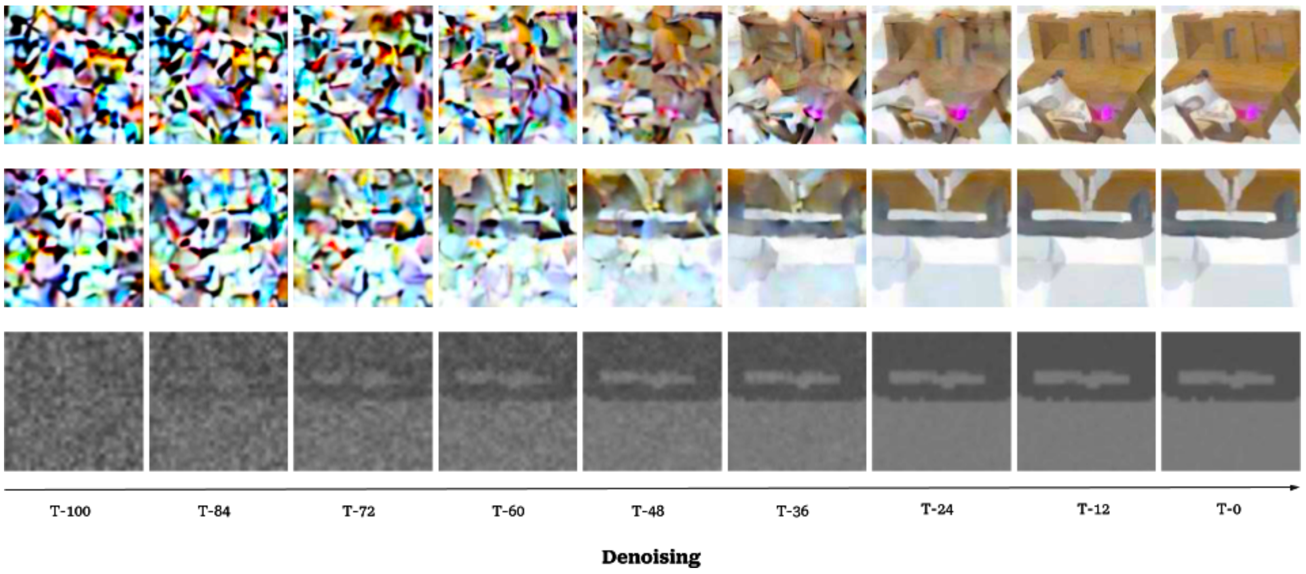


Figure 5: An overview of our Denoising process. In Figure 5 and Figure 6, we give an example of the denoising process of one of our ablations, the Diffuser-2D model. This model utilizes the 2D autoencoder of (Rombach et al., 2022) with (Janner et al., 2022).

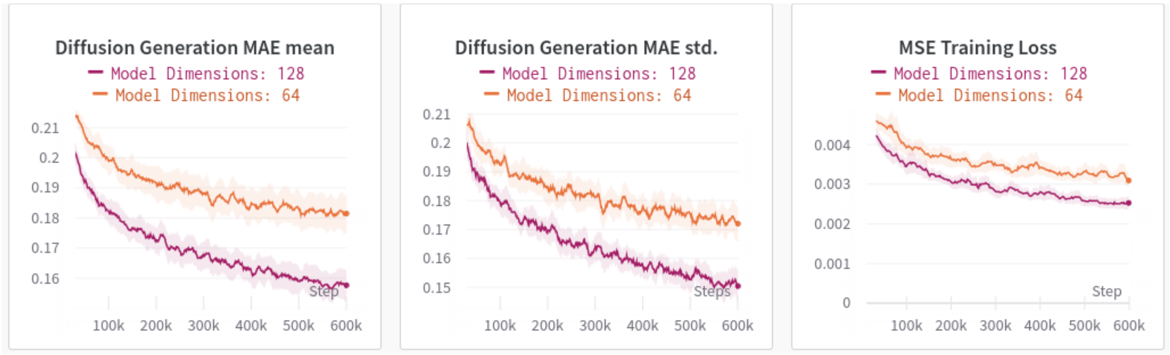


Figure 6: Diffusion Loss Comparison. Here we give study how varying the Diffusion model's size changes the performance of the model. As can be seen, scaling the model from 64 hidden dimensions to 128 strictly increases generation quality, and would likely follow scaling laws observed in (Kaplan et al., 2020).

F. Task Distribution

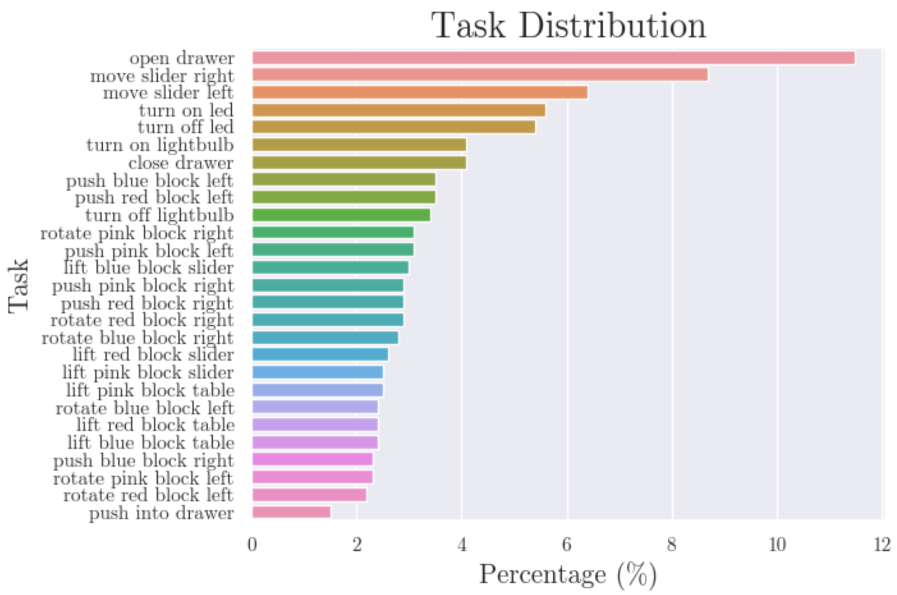
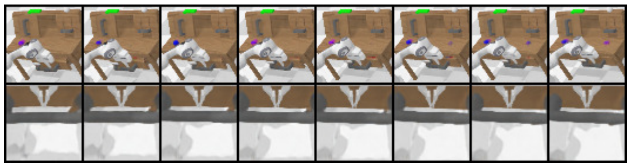


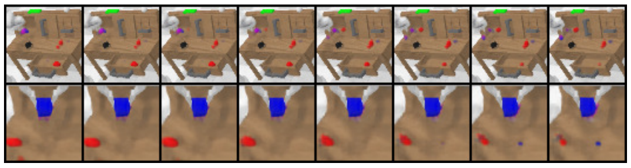
Figure 7: The Evaluation Task Distribution. We visualize the distribution of all the tasks considered in our experiments in Figure 7. Note the long-tailedness of this distribution, and how it skews evaluation scores upwards if one can solve the relatively easier tasks that occur most frequently, such as Open Drawer, Move Slider Right, and Move Slider Left. These tasks only deal with static objects, meaning there is very little generalization that is needed in order to solve these tasks when compared to other block tasks involving randomized block positions.

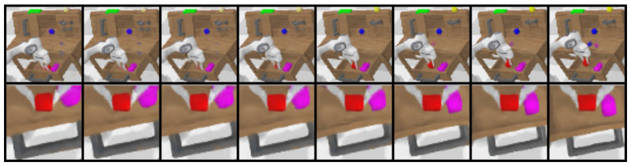
G. Representation Failures

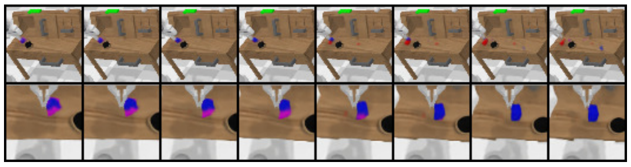
G.1. Diffuser-1D (β-TC VAE Latent Representation) Failures

We give a few failure cases of decoded latent plans, where the latent space is given by a trained from scratch β-TC VAE on the CALVIN D-D dataset. The top row of each plan comes from the static camera view, whilst the bottom one comes from the gripper camera view (a camera located at the tool center point of the robot arm). The VAE is trained by concatenating the images in the channel dimension, and compressing to 128 latent dimensions. Plans are sampled with a temporal stride of 9, such that each trajectory covers a total of 63 timesteps with $t = 0, 9, 18...63$. Interestingly, we found that replanning during rollout did not work, precluding the possibility of success on CALVIN with our implementation of this method.


(a) An example of the Close Drawer Task. Notice the flickering block on the top right of the table. Also not the entangled red and blue blocks at the top left of the table.

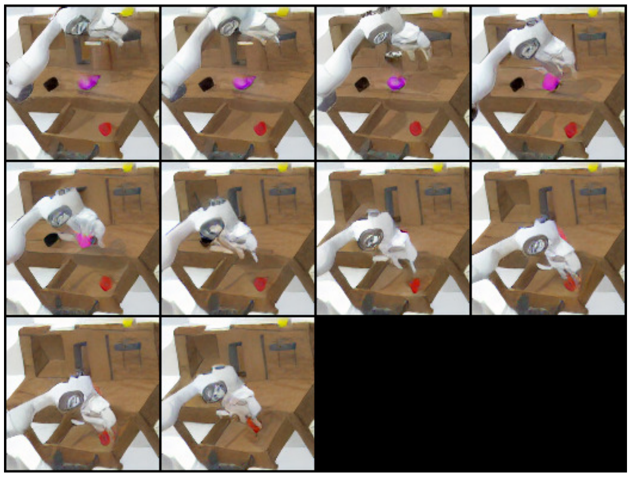

(b) An example of the Lift Blue Block Slider Task. The gripper view is temporally incoherent, red and blue blocks in slider are entangled.

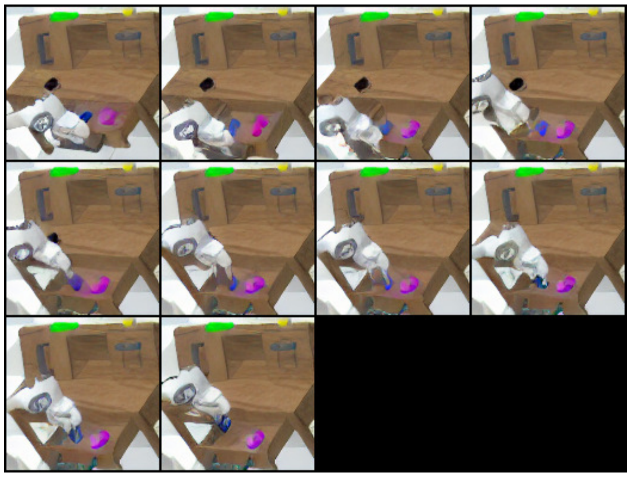

(c) An example of the Lift Red Block Drawer task. Two blocks begin to appear on the table at the end of generation. The red block is also not clearly generated in the first frame.

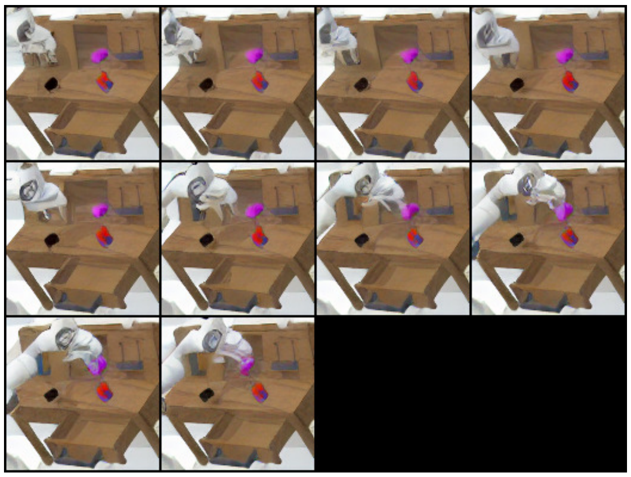

(d) An example of the Push Blue Block Right task. The blue block on the table becomes red by the end of the static view, whereas the opposite happens in the gripper view.

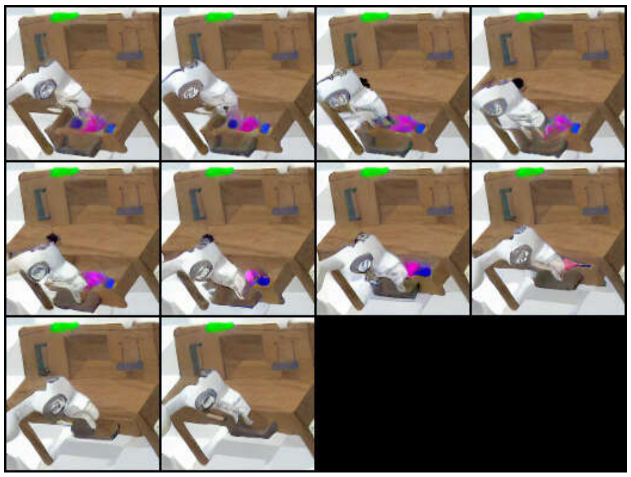
G.2. Diffuser-2D (Stable Diffusion Latent Representation) Failures

We additionally give some failure cases for Diffuser-2D. For more information on the training of this model, please refer to Appendix E. We also found that replanning during rollout did not work with this model.


(a) An example of the Lift Red Block Drawer Task. Note the pink block that disappears.

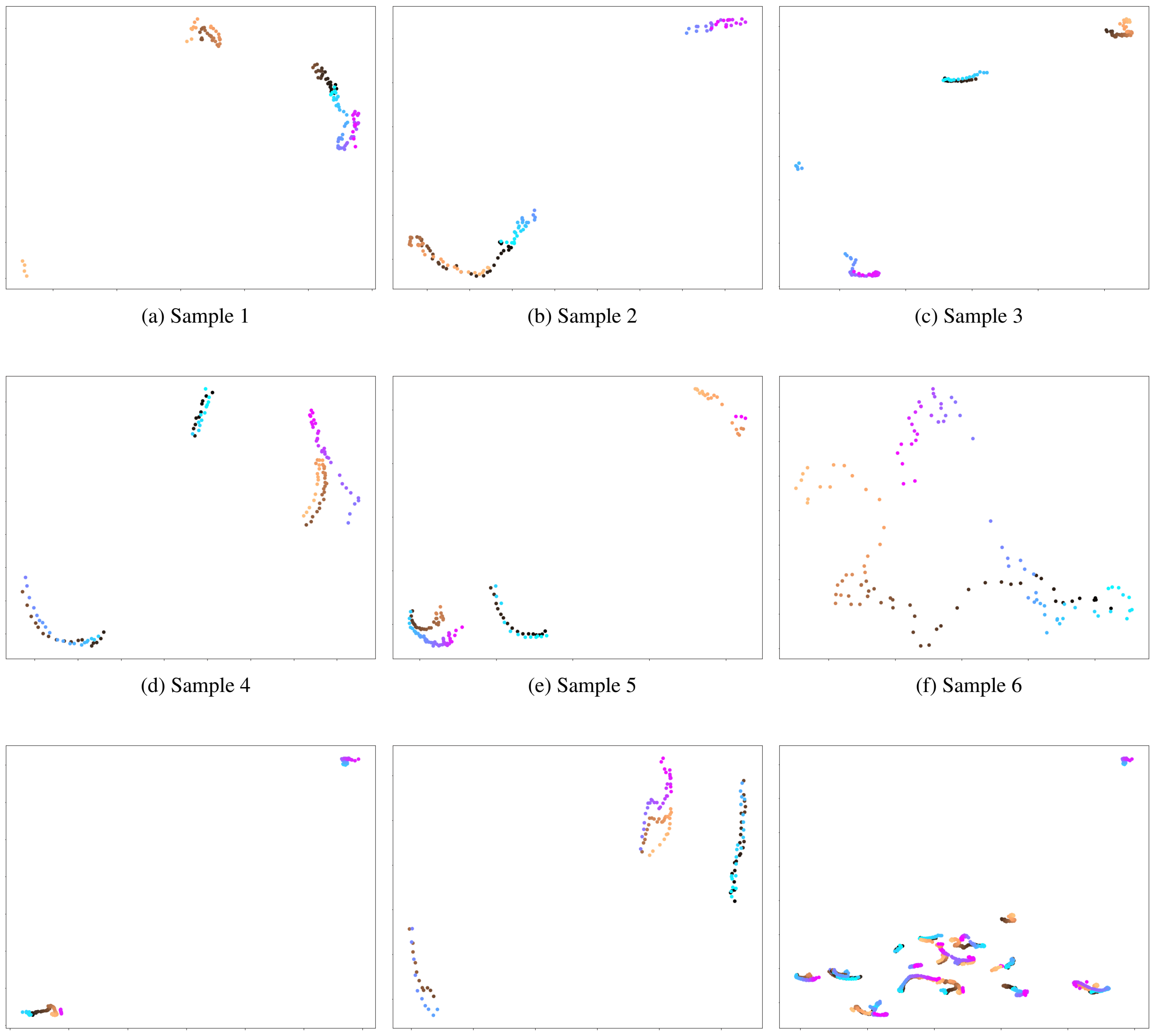

(b) An example of the Lift Blue Block Drawer Task. The gripper arm is entangled with the block.


(c) An example of the Lift Pink Block Slider Task. Note the entangled red/blue blocks.


(d) An example of the Close Drawer Task. Note the entangled pink/blue blocks.

H. TSNE Comparison between Groud Truth (GT) trajectory and Diffuser-1D (DM) trajectory

In order to better understand whether the representation failures found in Appendix G are a result of the underlying encoder or the diffusion model, we visualize the TSNE embeddings of an encoded successful trajectory from the dataset, which we refer to as a Ground Truth trajectory, and the TSNE embeddings of generated trajectories from Diffuser-1D (DM) in Figure 10. If we observe that the DM's embedddings are fairly close to the GT-VAE's, then we can reasonably presume that the VAE is the failure mode, whereas if the trajectories are wildly different this would imply that the DM is failing to model the VAE's latent distribution properly. Here, all samples other than 6 appear to fairly close, so we suspect that the failure case lies in the underlying latent distribution and not the DM's modeling capabilities. This is further backed by LCD, as we show that by using the proper underlying latent space with a LLP leads to success.



(a) Sample 1 (b) Sample 2 (c) Sample 3

(d) Sample 4 (e) Sample 5 (f) Sample 6

(g) Sample 7 (h) Sample 8 (i) All

Figure 10: TSNE visualization of GT-VAE trajectory vs. Diffuser-1D trajectory, where the purple and light blue color range is the ground truth VAE, and the copper color range is Diffuser-1D. All states are normalized, and all trajectories are taken from the task "lift pink block table".

I. HULC Latent Plan TSNE

We give TSNE emmbeddings of several Latent Plans generated during inference by HULC below.

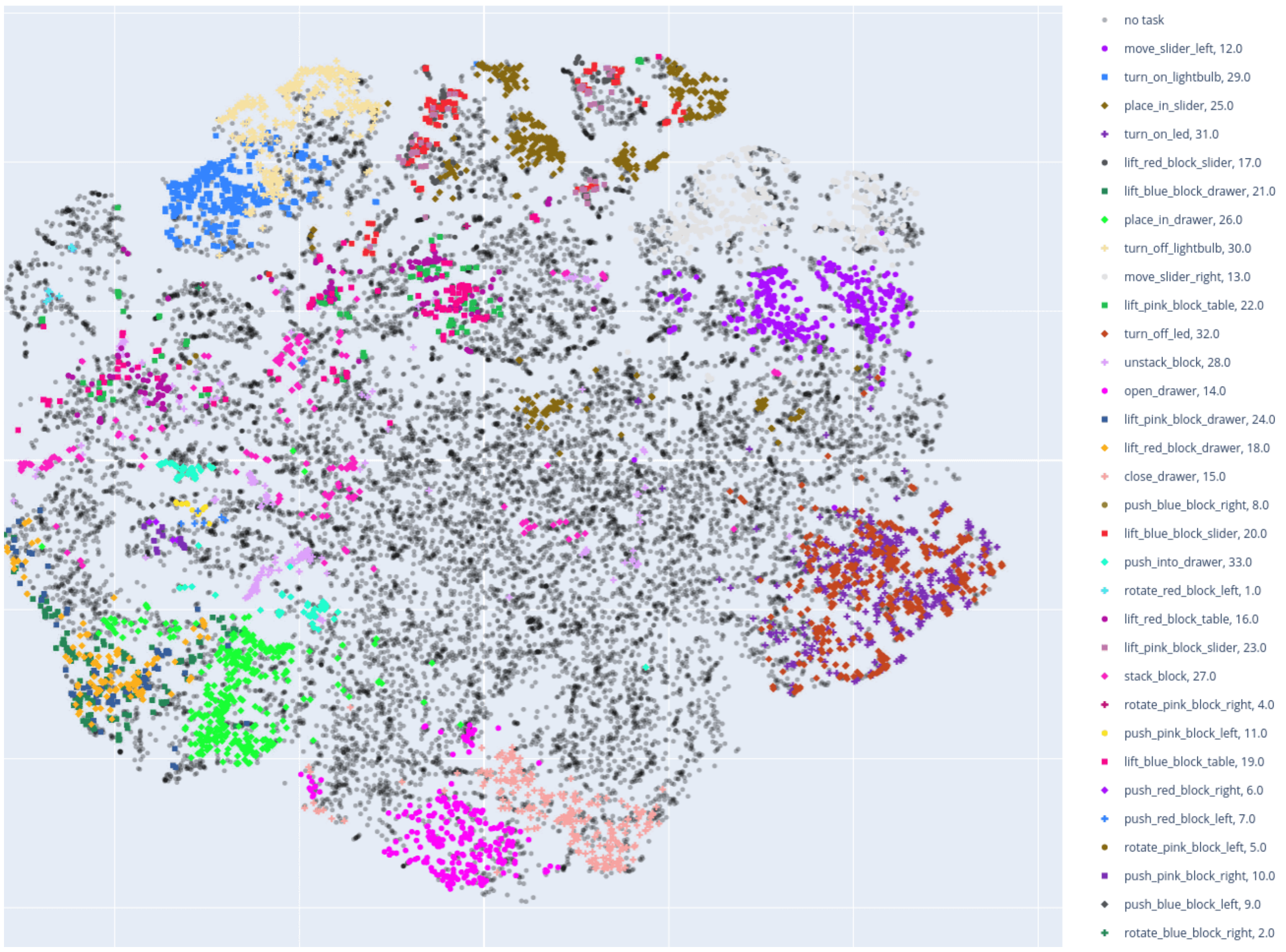


Figure 11: TSNE of Latent Plan. We give a TSNE embedding of the latent plan space of HULC in Figure 11. The latent plan space is the communication layer between the high level policy and low level policy of the HULC model, which corresponds to the intermediate layer between the lower level and lowest level policy in our method. We clarify that this is not the latent goal space that our model does generation in. Our method performs latent generation in the earlier layer from the output of the goal encoder, which corresponds to 32 latent dimensions.