Diffusion Subgoal Planning for Long-Horizon Offline Goal-Conditioned Reinforcement Learning

Hengrui Zhang[1] **Yuhu Cheng**[1] **C. L. Philip Chen**[2] **Xuesong Wang**[1]*

[1]School of Information and Control Engineering, China University of Mining and Technology
[2]School of Computer Science and Engineering, South China University of Technology

{hengruizhang, chengyuhu, wangxuesong}@cumt.edu.cn
{philip.chen}@ieee.org

Abstract

Offline goal-conditioned reinforcement learning (GCRL) learns goal-directed policies from reward-free data, but in long-horizon tasks, goal-conditioned value functions often provide unstable guidance due to sparse rewards and discounting. Hierarchical methods partially mitigate this issue via subgoal decomposition; however, high-level decision-making still relies on noise-sensitive value estimates, leading to unstable behavior in complex environments. We address this limitation by proposing **D**iffusion **S**ubgoal **P**lanning (**DSP**), a diffusion-based framework for high-level subgoal generation. DSP casts high-level planning as guided generative inference over goal-conditioned subgoals and learns both conditional and unconditional flows, enabling classifier-free guidance to introduce a goal-directed bias at inference time. By removing explicit value-based guidance from high-level planning, DSP generates reachable and goal-directed subgoals through a generative model while retaining hierarchical execution. Experiments on offline GCRL benchmarks demonstrate that DSP outperforms prior methods on a range of navigation and manipulation tasks, with particularly strong performance in maze environments that require multi-step subgoal planning.

Project page: https://henry0132.github.io/DSP

Repository: https://github.com/Henry0132/DSP

1 Introduction

Goal-conditioned reinforcement learning (GCRL) formulates decision-making problems in terms of desired outcomes, reducing the need to manually design reward functions in high-dimensional spaces and enabling task generalization [16, 43, 18]. Offline GCRL [57, 63] further extends this setting to learning solely from reward-free trajectories, resembling self-supervised learning from past experience. However, without explicit rewards, desired goals are often temporally distant, and value function estimation can suffer from limited data coverage and accumulated errors [55, 17, 32], resulting in noisy and unreliable learning signals in long-horizon tasks.

Hierarchical Implicit Q-Learning (HIQL) [55] addresses long-horizon challenges by introducing a two-level policy, where a high-level policy proposes intermediate subgoals and a low-level policy executes actions conditioned on them. While this hierarchy improves stability over flat policies, large-scale offline GCRL evaluations show that in tasks involving complex navigation, spatial search, or high-dimensional manipulation, HIQL can struggle to obtain reliable guidance from noisy value functions, leading to degraded performance [57, 1, 53]. Subsequent analyses attribute these failures

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

in part to imprecise value guidance and the resulting errors in high-level planning [58, 1]. Since hierarchical performance depends critically on subgoal quality, several methods seek to improve value estimation itself, such as Pi-HIQL [17], which enforces geometric structure via Eikonal regularization, and OTA [1], which promotes hierarchical consistency through option-aware mechanisms.

However, these approaches share a common assumption that high-level planning should be driven by value function estimates. When subgoals are selected through noisy values, long-horizon planning can remain unstable. This naturally raises a central question:

Can hierarchical planning be retained without value-based high-level guidance?

Diffusion models [23, 66] provide a promising alternative to value-based planning by generating reachable intermediate states that reflect the structure of the offline data distribution [27, 2]. Moreover, goal-directed preferences can be incorporated into the sampling process via guidance mechanisms such as classifier-free guidance (CFG) [22], enabling controllable subgoal generation without explicit high-level value estimation.

Motivated by these properties, we propose **D**iffusion **S**ubgoal **P**lanning (**DSP**), a diffusion-based framework for high-level subgoal generation in offline GCRL. DSP formulates high-level planning as guided generative inference over goal-conditioned subgoals, enabling high-level decision-making without explicit value-based guidance while preserving the benefits of hierarchical execution. Concretely, DSP learns both conditional and unconditional velocity fields, and uses classifier-free guidance at inference time to bias subgoal generation toward the desired goal. This guidance mechanism further admits an implicit advantage-weighted interpretation at the subgoal level. Experiments on offline GCRL benchmarks demonstrate that DSP outperforms prior methods on a range of navigation and manipulation tasks, with particularly strong gains in long-horizon maze environments that require multi-step subgoal planning.

2 Related Works

Offline GCRL. Offline goal-conditioned reinforcement learning aims to learn a universal policy that reaches arbitrary target states from arbitrary initial states using a fixed dataset [60, 16, 52, 57]. Most approaches treat goals as future states and adopt self-supervised training schemes, including hindsight relabeling [3, 11], state occupancy matching [63, 10], and related goal relabeling methods [51, 67]. Recent work further incorporates hierarchical structures to decompose complex tasks into multi-step plans via recursive subgoal inference [55, 17, 1, 28, 35]. Despite differences in formulation, these methods share a common assumption that high-level subgoals are optimized using learning signals derived from goal-conditioned value functions, making noisy value estimates a central bottleneck for reliable long-horizon planning in offline settings.

Hierarchical RL. Hierarchical reinforcement learning introduces decision-making at multiple temporal scales to address long-horizon planning and exploration [19, 5, 15]. Early approaches rely on graph-based subgoal planning [75, 30], while later methods generate intermediate waypoints using temporal distance [44], latent abstractions [9, 15], or value-based criteria [1, 62]. Although these approaches aim to simplify planning by defining reachable high-level subgoals, graph-based methods are often computationally expensive and difficult to scale [61, 34, 4], while waypoint prediction methods that depend on value functions or distance estimates are highly sensitive to estimation accuracy, particularly under sparse data coverage [10]. As a result, existing hierarchical RL methods can remain sensitive to value-function noise, particularly in offline settings [57].

Diffusion Policies for RL. Diffusion models have been adopted in reinforcement learning as expressive generative models [23, 64–66, 40, 45], with applications to action generation [70, 47, 7, 20], trajectory modeling [2, 39], and goal-conditioned or planning-oriented generation [26, 29, 73]. Recent methods further use guidance or flow matching to improve action-level policy learning and controllability [13, 59]. Other works explore hierarchical diffusion-based planning or subgoal generation [38, 6, 74], where diffusion models are typically used as trajectory or milestone planners [25, 24], often combined with MPC, return guidance, or multi-step planning procedures. In contrast, DSP uses guided generation as a high-level subgoal decision mechanism in hierarchical offline GCRL, replacing explicit value-based subgoal selection while retaining a separately trained low-level executor.

3 Preliminaries

Problem Setting. Offline GCRL is formulated as a finite-horizon discounted Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{G}, \mathcal{P}, r, \mathcal{P}_g, d_0, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{G}$ denote the state, action, and goal spaces, respectively. The transition dynamics are given by $\mathcal{P}(s' \mid s, a)$, and the reward function is $r(s, g)$. The initial state distribution d_0 and goal distribution $\mathcal{P}_g$ specify how episodes are initialized. The discount factor is $\gamma \in [0, 1)$. Following standard practice, we assume $\mathcal{G} = \mathcal{S}$. At the beginning of each episode, a goal $g \sim \mathcal{P}_g$ is sampled, and the agent aims to reach g by maximizing the expected discounted return $J(\pi) = \mathbb{E}_{\tau \sim \pi(\cdot|g)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, g) \right]$, where $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$ is a trajectory induced by a goal-conditioned policy $\pi(a \mid s, g)$. The corresponding goal-conditioned value function is $V^\pi(s, g) = \mathbb{E}_{\tau \sim \pi(\cdot|g)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, g) \;\middle|\; s_0 = s \right]$. In the offline setting, the policy is learned solely from a fixed dataset $\mathcal{D}$ of trajectories, without further interaction with the environment.

Hierarchical Implicit Q-Learning. In GCRL, accurately estimating value functions for distant goals is particularly challenging in long-horizon tasks. HIQL [55] addresses this issue by introducing a hierarchical policy structure built on implicit Q-learning [33], which directly learns value estimates from offline data. Specifically, HIQL learns a goal-conditioned value function by minimizing the following expectile regression objective:

$$\mathcal{L}_V = \mathbb{E}_{(s_t, s_{t+1}) \sim \mathcal{D},\, g \sim p^{\mathcal{D}}} \left[L_2^\varepsilon \left(r(s_t, g) + \gamma \bar{V}(s_{t+1}, g) - V(s_t, g) \right) \right], \tag{1}$$

where $p^{\mathcal{D}}$ denotes the goal sampling distribution, $\bar{V}$ is a target value network, and L_2^ε is the expectile loss

$$L_2^\varepsilon(u) = |\varepsilon - \mathbf{1}(u < 0)|\, u^2, \tag{2}$$

with expectile parameter ε. Following prior work [71, 51, 72], the reward is defined as $r(s_t, g) = -\mathbf{1}(s_t \neq g)$. HIQL decomposes the policy into two levels. The high-level policy $\pi^h(s_{t+k} \mid s_t, g)$ predicts a subgoal k steps ahead, while the low-level policy $\pi^l(a_t \mid s_t, s_{t+k})$ executes actions to reach the proposed subgoal. Both policies are trained using advantage-weighted regression (AWR):

$$\mathcal{J}(\pi^h) = \mathbb{E}_{(s_t, s_{t+k}) \sim \mathcal{D},\, g \sim p^{\mathcal{D}}} \left[\exp\left(\beta^h A^h(s_t, s_{t+k}, g) \right) \log \pi^h(s_{t+k} \mid s_t, g) \right], \tag{3}$$

$$\mathcal{J}(\pi^l) = \mathbb{E}_{(s_t, a_t, s_{t+1}, s_{t+k}) \sim \mathcal{D}} \left[\exp\left(\beta^l A^l(s_t, s_{t+1}, s_{t+k}) \right) \log \pi^l(a_t \mid s_t, s_{t+k}) \right], \tag{4}$$

where β^h and β^l are inverse temperature parameters, and the advantages are defined as $A^h(s_t, s_{t+k}, g) = V(s_{t+k}, g) - V(s_t, g)$ and $A^l(s_t, s_{t+1}, s_{t+k}) = V(s_{t+1}, s_{t+k}) - V(s_t, s_{t+k})$.

4 High-Level Planning with Reduced Horizon-Dependent Value Noise

This section analyzes a limitation of HIQL in long-horizon settings and motivates our approach. We first use a one-dimensional illustrative example to show how hierarchical structures improve the signal-to-noise ratio (§4.1). We then provide a theoretical analysis showing that DSP removes one horizon-dependent error path caused by noisy high-level value estimates while preserving the benefits of hierarchy (§4.2). Finally, we empirically validate these insights on representative offline GCRL benchmarks (§4.3).

4.1 Why Value-Function Noise Limits Long-Horizon Subgoal Planning

To analyze how value-function noise constrains long-horizon subgoal planning, we revisit the one-dimensional toy example introduced in [55]. As shown in Figure 1, the environment consists of a one-dimensional state space, where the agent can move left or right at each time step toward a goal located at the rightmost position. The reward is sparse: $r(s_t = g) = 0$ and -1 otherwise. Assuming unit discount, the optimal goal-conditioned value function is $V^*(s, g) = -|s - g|$. We model the learned value function as a noisy approximation of the optimal one, $\hat{V}(s, g) = V^*(s, g) + \sigma z_{s,g} V^*(s, g)$, where $z_{s,g} \sim \mathcal{N}(0, I)$ and σ controls the noise scale. This formulation captures a common empirical challenge in offline GCRL: value estimation errors tend to grow with the distance to the goal.

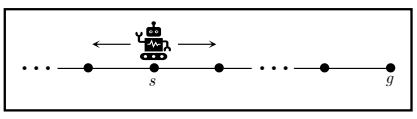


Figure 1: 1-D toy environment.

Under this setting, flat policies incur a higher probability of incorrect actions than hierarchical policies, as formalized in Proposition 4.1.

Proposition 4.1 (Hierarchical Policy Can Reduce Policy Error [55])**.** *In the toy environment, the probability of the flat policy π selecting an incorrect action is*

$$\mathcal{E}(\pi) = \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{T^2+1}}\right), \tag{5}$$

and the probability of the hierarchical policy $\pi^l \circ \pi^h$ selecting an incorrect action is bounded by

$$\mathcal{E}(\pi^l \circ \pi^h) \leq \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{(T/k)^2+1}}\right) + \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{k^2+1}}\right), \tag{6}$$

where Φ denotes the cumulative distribution function of the standard normal distribution, $\Phi(x) = \mathbb{P}[z \leq x] = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}dt$.

Proposition 4.1 shows that hierarchy improves robustness by reducing the effective decision horizon from T to T/k. However, high-level planning in HIQL still depends on value estimates $V(s, g)$, causing estimation errors to propagate through subgoal selection with a T/k scaling. Consequently, hierarchy attenuates but does not eliminate horizon-dependent value noise, which can become a bottleneck in long-horizon tasks.

4.2 Diffusion Subgoal Planning Reduces Horizon-Dependent Value Noise

We analyze the error structure of DSP in the same toy environment to contrast its behavior with HIQL. DSP adopts a two-level hierarchical policy: the high-level policy π^h_{DSP} generates a subgoal, and the low-level policy π^ℓ_{DSP} produces an action conditioned on that subgoal. The overall policy is therefore composed as $\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}$.

Appendix B.2 first derives the exact one-step decision error for an arbitrary generated subgoal distribution, without assuming Gaussianity or unimodality. To obtain a transparent closed-form comparison with HIQL, Proposition 4.2 considers a matched-distance setting in which both methods select subgoals at the same distance k. This controls for low-level execution difficulty and isolates the difference between generative and value-based high-level direction selection.

Proposition 4.2 (Reduced High-Level Value-Noise Amplification)**.** *Consider the one-dimensional environment illustrated in Figure 1, with current state s, final goal $g = s + T$, and $1 \leq k < T$. Suppose that both DSP and HIQL select one of two subgoals: $s + k$ in the correct direction or $s - k$ in the incorrect direction. Let ϵ denote the probability that DSP selects $s - k$, and define*

$$p_\ell(k) := \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{k^2+1}}\right), \quad p_h(T,k) := \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{(T/k)^2+1}}\right). \tag{7}$$

Here, $p_\ell(k)$ is the probability that the low-level policy moves away from its selected subgoal at distance k, while $p_h(T, k)$ is the probability that HIQL selects the wrong-direction subgoal.

Under the independent Gaussian value-noise model of Proposition 4.1, and assuming that DSP's generated direction is independent of the low-level value noise, the exact one-step decision errors are

$$\mathcal{E}\left(\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}\right) = p_\ell(k) + (1 - 2p_\ell(k))\,\epsilon, \tag{8}$$

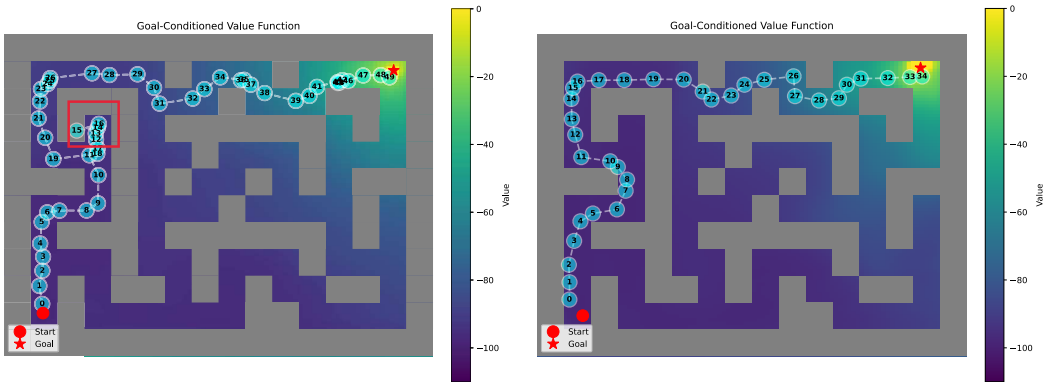
$$\mathcal{E}\left(\pi^\ell \circ \pi^h\right) = p_\ell(k) + (1 - 2p_\ell(k))\,p_h(T,k). \tag{9}$$

Consequently,

$$\mathcal{E}\left(\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}\right) < \mathcal{E}\left(\pi^\ell \circ \pi^h\right) \quad \Longleftrightarrow \quad \epsilon < p_h(T,k). \tag{10}$$

For the additional closed-form special case in which σ_{data} is independent of T and the wrong-direction probability is parameterized as

$$\epsilon = \Phi\left(-\frac{k}{\sigma_{\mathrm{data}}}\right), \tag{11}$$



(a) HIQL$^{\mathrm{w/o}}$ (b) DSP

Figure 2: Comparative visualization of high-level planning in the `antmaze-giant-navigate-v0` environment. The state space is projected onto the 2D (x, y) plane, where the background heatmap illustrates the landscape of the goal-conditioned value function; brighter regions correspond to higher values. HIQL$^{\mathrm{w/o}}$ selects subgoals through value-based guidance, making its subgoal trajectories sensitive to value-function noise. In contrast, DSP generates smoother and topologically connected paths without explicit value-based high-level guidance.

condition (10) *is equivalent to*

$$T > k\sqrt{\left(\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k}\right)^2 - 1} \tag{12}$$

when $\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k} > 1$. *When* $\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k} \leq 1$, *condition* (10) *holds for every* $T > 0$.

Remark. The matched-distance setting assigns DSP and HIQL the same local subgoal distance and therefore the same low-level error $p_\ell(k)$. Their ordering is determined solely by whether DSP's wrong-direction probability ϵ is smaller than HIQL's high-level value-comparison error $p_h(T, k)$. As T/k increases, $p_h(T, k) \to \frac{1}{2}$, and Equation (9) consequently approaches $\frac{1}{2}$, reflecting the degradation of long-horizon value-based direction selection under the noisy-value model.

The Gaussian expression in Equation (11) is only a closed-form special case for parameterizing ϵ. The general derivation in Appendix B.2 allows the generated subgoal distribution to be non-Gaussian, multimodal, and dependent on T. Accordingly, Proposition 4.2 does not imply horizon-independent DSP error: the generated subgoal distribution may deteriorate as the task horizon increases or offline data coverage decreases. The result instead isolates the removal of the explicit T/k-scaled high-level value-noise path present in HIQL-style subgoal selection.

The proof of Proposition 4.2 is provided in Appendix B.2, and empirical validation is presented in Section 4.3.

4.3 Empirical Evidence: DSP Generates More Reliable Subgoals

We qualitatively compare the high-level subgoals generated by HIQL$^{\mathrm{w/o}}$[2] and DSP on `antmaze-giant-navigate-v0`, a representative high-dimensional offline GCRL task requiring long-distance navigation through a complex maze under sparse rewards. This environment provides a challenging test for the stability of high-level planning over long horizons.

Figure 2 overlays the subgoal trajectories produced by HIQL$^{\mathrm{w/o}}$ and DSP on the same value landscape. As shown in Figure 2(a), HIQL$^{\mathrm{w/o}}$ often proposes subgoals that become trapped in local optima or intersect obstacles, resulting in unstable plans. These behaviors are consistent with our analysis in

[2]HIQL$^{\mathrm{w/o}}$ removes the subgoal representation bottleneck in HIQL and operates directly on state-space subgoals. As shown in Table 1, HIQL and HIQL$^{\mathrm{w/o}}$ achieve comparable performance on `antmaze-giant-navigate-v0`, enabling a fair qualitative comparison.

Sections 4.1 and 4.2, where high-level planning based on noisy value estimates exhibits horizon-dependent error accumulation.

In contrast, DSP produces smoother subgoal sequences that respect the maze topology and maintain progress toward the goal. By generating subgoals through a diffusion-based process rather than value comparison, DSP reduces sensitivity to local errors in the value landscape. As a result, the agent more reliably follows feasible corridors in this long-horizon navigation task. Additional visualizations on mazes of varying sizes are provided in Appendix O.

5 Diffusion Subgoal Planning

This section introduces Diffusion Subgoal Planning (DSP). DSP aims to retain the execution efficiency and local guidance benefits of hierarchical structures while removing explicit value-based guidance from high-level subgoal generation. Instead, subgoals are generated through a controllable diffusion-based process grounded in the data distribution. Section 5.1 formulates subgoal planning as probabilistic inference, Section 5.2 describes the learning of goal-conditioned velocity fields, and Section 5.3 presents inference-time control via classifier-free guidance and its connection to implicit advantage-weighted subgoal selection. Section 5.4 summarizes the complete algorithm.

Notation: Unless otherwise specified, superscripts index diffusion time steps $i \in [0, 1]$, while subscripts index trajectory time steps $t \in \{0, \dots, T-1\}$.

5.1 Subgoal Generation as Probabilistic Inference

Hierarchical goal-conditioned reinforcement learning can be formulated as a probabilistic inference problem. At the trajectory level, we introduce a binary optimality variable $\mathcal{O}$ as an analytical device indicating whether a trajectory reaches the final goal with high return. Under this formulation, high-level planning corresponds to sampling subgoals from the posterior

$$w \sim p(w \mid s, \mathcal{O} = 1). \tag{13}$$

This posterior characterizes subgoals that are likely to support goal-reaching behavior from the current state s.

Existing approaches typically approximate this posterior by first learning a goal-conditioned value function $V(s, g)$ and then deriving the high-level policy via value-based optimization [55, 5, 1]. As analyzed in Section 4.1, this approximation couples subgoal selection to value estimation errors, leading to horizon-dependent noise amplification and brittle long-horizon planning.

We instead adopt a generative planning perspective. By Bayes' rule, the posterior can be decomposed as

$$\log p(w \mid s, \mathcal{O} = 1) \propto \log p(w \mid s) + \log p(\mathcal{O} = 1 \mid w, s). \tag{14}$$

Here, $\log p(w \mid s)$ defines a data-induced prior over reachable and feasible subgoals, while $\log p(\mathcal{O} = 1 \mid w, s)$ captures a goal-directed preference. In DSP, this preference is operationalized through the goal condition g: the goal-conditioned branch captures how the subgoal distribution changes when the desired goal is specified. Rather than explicitly regressing this preference term through value functions, DSP learns unconditional and goal-conditioned generative velocity fields. Their difference implicitly encodes goal preference at inference time, enabling subgoal sampling that is biased toward goal-relevant waypoints without value-based high-level planning and its associated horizon-dependent noise.

5.2 Learning the Goal-Directed Velocity Field

We model subgoal generation as a deterministic flow that transforms a simple prior noise variable $w^0 \sim \mathcal{N}(0, I)$ into a goal-conditioned subgoal distribution $p(w \mid s, g)$ via a parameterized velocity field. Rather than predicting noise or scores, we adopt a flow matching objective that directly learns the velocity field defining the generative ordinary differential equation, enabling stable and efficient training [48, 49, 46].

Given a training subgoal w^1, constructed as a k-step future state from the offline dataset, i.e., $w^1 = s_{t+k}$, we define a linear interpolation path

$$w^i = (1 - i)w^0 + iw^1, \quad i \in [0, 1], \tag{15}$$

and supervise the model to match the corresponding velocity along this path, $\frac{dw^i}{di} = w^1 - w^0$. This objective encourages the learned flow to transport samples from the prior toward the empirical subgoal distribution.

To jointly learn the unconditional distribution $p(w \mid s)$ and the goal-conditioned distribution $p(w \mid s, g)$ within a single model, we employ CFG through conditional dropout during training. The resulting objective is

$$\mathcal{L}_{\mathrm{DSP}}(\theta) = \mathbb{E}_{\substack{i\sim\mathcal{U}(0,1),\ w^0\sim\mathcal{N}(0,I),\\ (s_t,w^1)\sim\mathcal{D},\ g\sim p^{\mathcal{D}}}} \left[\left\| v_\theta(w^i, i, s_t, \tilde{g}) - (w^1 - w^0) \right\|^2 \right], \tag{16}$$

where the conditioning variable $\tilde{g}$ equals g with probability $1 - p_{\mathrm{unc}}$ and is set to $\emptyset$ otherwise. This design enables a single network to represent both $v_\theta(\cdot \mid s, g)$ and $v_\theta(\cdot \mid s)$, laying the foundation for controllable guided sampling at inference time.

5.3 Guided Sampling as Implicit Policy Extraction

Although flow matching learns the empirical goal-conditioned subgoal distribution during training, DSP further exploits the relationship between the unconditional and conditional distributions at inference time. Specifically, classifier-free guidance performs a density-ratio reweighting of the behavior-induced subgoal distribution.

Let $v_\theta(w^i, i, s, \emptyset)$ and $v_\theta(w^i, i, s, g)$ denote the unconditional and goal-conditioned velocity fields. DSP constructs

$$\hat{v}_\theta(w^i, i) = v_\theta(w^i, i, s, \emptyset) + \omega \left(v_\theta(w^i, i, s, g) - v_\theta(w^i, i, s, \emptyset) \right), \tag{17}$$

where ω controls the strength of goal-directed guidance. Under the score approximation interpretation of CFG, the conditional-unconditional difference corresponds to $\nabla_w \log \frac{p(w|s,g)}{p(w|s)}$.

The following proposition establishes the connection between this density ratio and the RL advantage of the induced high-level behavior policy.

Proposition 5.1 (CFG as Implicit Relative-Advantage-Weighted Subgoal Extraction)**.** *Let $\pi_\beta(w|s) = p(w|s)$ denote the behavior-induced subgoal distribution. Assume that the goal-conditioned subgoal distribution is obtained by future-state relabeling under π_β. Then,*

$$\frac{p(w|s,g)}{p(w|s)} = \frac{Q_\beta(s,w,g)}{V_\beta(s,g)}, \tag{18}$$

where $Q_\beta(s,w,g)$ is the goal-conditioned value of selecting subgoal w under π_β, and $V_\beta(s,g) = \mathbb{E}_{w\sim\pi_\beta(\cdot|s)}[Q_\beta(s,w,g)]$.

Equivalently,

$$\log \frac{p(w|s,g)}{p(w|s)} = \log \left(1 + \frac{A_\beta(s,w,g)}{V_\beta(s,g)} \right), \tag{19}$$

where $A_\beta(s,w,g) = Q_\beta(s,w,g) - V_\beta(s,g)$ is the behavior-policy RL advantage.

Therefore, CFG sampling induces the tilted distribution

$$p_\omega(w \mid s, g) \propto \pi_\beta(w \mid s) \left(\frac{Q_\beta(s,w,g)}{V_\beta(s,g)} \right)^\omega \propto \pi_\beta(w \mid s) Q_\beta(s,w,g)^\omega. \tag{20}$$

Equivalently,

$$p_\omega(w \mid s, g) \propto \pi_\beta(w \mid s) \exp\left(\omega \widetilde{A}_\beta(s,w,g) \right), \tag{21}$$

where $\widetilde{A}_\beta(s,w,g) = \log \frac{Q_\beta(s,w,g)}{V_\beta(s,g)}$. This corresponds to implicit relative-advantage-weighted subgoal extraction.

Proposition 5.1 clarifies that the implicit quantity amplified by CFG is the log-relative advantage $\widetilde{A}_\beta(s,w,g) = \log \frac{Q_\beta(s,w,g)}{V_\beta(s,g)}$, which is a strictly increasing transformation of the additive RL advantage. Therefore, CFG preserves the ordering of subgoals induced by $A_\beta(s,w,g)$ while avoiding explicit value estimation during high-level planning. The detailed proof of Proposition 5.1 is provided in Appendix B.3.

5.4 Practical Algorithm

We combine the diffusion-based high-level planner with the value function and low-level policy learning scheme of HIQL, using the following objectives:

$$\mathcal{L}_V(\phi) = \mathbb{E}_{(s_t, s_{t+1}) \sim \mathcal{D},\, g \sim p^{\mathcal{D}}} \left[L_2^{\varepsilon} \left(r(s_t, g) + \gamma V_{\bar{\phi}}(s_{t+1}, g) - V_\phi(s_t, g) \right) \right]. \tag{22}$$

$$\mathcal{J}_{\pi^\ell}(\vartheta) = \mathbb{E}_{(s_t, a_t, s_{t+1}, w) \sim \mathcal{D}} \left[\exp\left(\alpha \left(V_\phi(s_{t+1}, w) - V_\phi(s_t, w) \right) \right) \log \pi^\ell_\vartheta(a_t \mid s_t, w) \right]. \tag{23}$$

Algorithms 1 and 2 summarize the training and inference procedures of DSP. The high-level planner generates subgoals via guided diffusion over velocity fields, while the low-level policy executes actions using value-based advantage-weighted updates.

Algorithm 1 DSP Training

Input: Dataset $\mathcal{D}$; value $V_\phi, V_{\bar{\phi}}$; flow v_θ; policy π^ℓ_ϑ.
while not converged **do**
  Sample $(s_t, a_t, s_{t+1}, w, g) \sim \mathcal{D}$
  Sample $w^0 \sim \mathcal{N}(0, I)$, $i \sim \mathcal{U}(0, 1)$
  Set $w^1 \leftarrow w$, $w^i \leftarrow (1 - i)w^0 + iw^1$
  Update V_ϕ using Eq. (22)
  $\bar{\phi} \leftarrow (1 - \kappa)\bar{\phi} + \kappa\phi$
  Update v_θ using Eq. (16)
  Update π^ℓ_ϑ using Eq. (23)
end while
Output: Trained V_ϕ, v_θ, and π^ℓ_ϑ

Algorithm 2 DSP Inference

Input: Goal g; state s_0; flow v_θ; policy π^ℓ_ϑ; guidance ω; steps N.
Define: $\Delta v_\theta(w, i, s, g) = v_\theta(w, i, s, g) - v_\theta(w, i, s, \emptyset)$
Initialize $s_t \leftarrow s_0$
while not done **do**
  Sample $w \sim \mathcal{N}(0, I)$
  for $i \in \{0, \frac{1}{N}, \ldots, \frac{N-1}{N}\}$ **do**
    $w \leftarrow w + \frac{1}{N}[v_\theta(w, i, s_t, \emptyset) + \omega \Delta v_\theta(w, i, s_t, g)]$
  end for
  $a_t \sim \pi^\ell_\vartheta(\cdot \mid s_t, w)$
  Execute a_t and observe s_{t+1}, $s_t \leftarrow s_{t+1}$
end while

6 Experiments

We evaluate DSP on OGBench, a challenging offline GCRL benchmark designed to test long-horizon reasoning and multi-goal composition. Section 6.1 describes the experimental setup. Sections 6.2 and 6.3 report results on locomotion and manipulation tasks, respectively. Implementation details and hyperparameters are provided in Appendices C and D. Hyperparameter sensitivity, high-level planner ablations, inference-time analysis, and stitching experiments are reported in Appendices I, J.1, K, and L, respectively.

6.1 Experimental Setup

We evaluate DSP on a subset of environments and datasets from OGBench, a benchmark designed for offline GCRL with multiple goal-conditioned evaluation protocols. OGBench includes datasets of varying characteristics to assess long-horizon reasoning, trajectory stitching, and multi-goal composition. We focus on two categories of tasks.

Locomotion. Locomotion tasks require controlling a robot to navigate mazes and reach target locations. We consider `point`, `ant`, and `humanoid` embodiments across `medium`, `large`, and `giant` maze layouts. We further include the challenging `antsoccer` task, where an ant robot must push a ball to a target location, evaluated under both `arena` and `medium` settings.

Manipulation. Manipulation tasks involve a 6-DoF robotic arm performing object-centric interactions, including cube grasping, button pressing, window opening, and drawer manipulation. These tasks evaluate the agent's ability to compose sequential behaviors under complex dynamics.

For comparison, we consider both flat and hierarchical baselines: **GCBC** [16], **CFGRL** [13], **GCIVL** [57], **OTA** [1], **Pi-HIQL** [17], and **HIQL** [55]. We follow the hyperparameter settings reported in the original works and rerun their released code to report mean and standard deviation. Additional details on OGBench and the baseline methods are provided in Appendices F and G.

6.2 Locomotion Results

We first evaluate DSP on the locomotion suite of OGBench. As summarized in Table 1, DSP achieves strong performance across the evaluated locomotion tasks, showing its effectiveness for offline

Table 1: Complete comparison between DSP and the offline GCRL baselines. The table reports the average binary success rate (%) across five test-time goals for each task, averaged over 5 seeds. Standard deviations are indicated by the ± symbol. Entries within 95% of the best-performing value in each row are highlighted in **bold**.

Datasets	Flat Policies			Hierarchical Policies				
	GCBC	CFGRL	GCIVL	OTA	Pi-HIQL	HIQL	HIQL$^{w/o}$	DSP
pointmaze-medium-navigate-v0	4.8 ± 6.9	65.2 ± 5.3	70.2 ± 5.9	85.4 ± 5.0	62.2 ± 7.3	70.6 ± 7.0	74.6 ± 6.1	**90.6** ± 3.8
pointmaze-large-navigate-v0	25.6 ± 6.3	74.6 ± 2.9	42.6 ± 5.3	87.6 ± 9.2	80.2 ± 13.3	39.4 ± 2.3	50.0 ± 10.4	**93.6** ± 2.9
pointmaze-giant-navigate-v0	2.2 ± 4.9	3.4 ± 2.5	1.0 ± 2.2	**69.4** ± 11.6	3.0 ± 2.9	5.0 ± 5.0	0.0 ± 0.0	43.4 ± 7.2
pointmaze-teleport-navigate-v0	23.8 ± 5.7	**49.2** ± 5.1	43.6 ± 2.5	40.2 ± 5.4	33.6 ± 9.0	8.8 ± 6.4	8.2 ± 6.2	**51.0** ± 5.8
antmaze-medium-navigate-v0	32.6 ± 8.5	34.8 ± 5.5	74.6 ± 8.0	91.6 ± 2.4	90.8 ± 2.7	93.2 ± 1.3	92.8 ± 2.4	**98.4** ± 0.5
antmaze-large-navigate-v0	23.0 ± 2.0	18.4 ± 5.7	15.0 ± 6.6	**89.2** ± 2.3	81.8 ± 3.1	**87.8** ± 1.5	**88.6** ± 2.1	**92.0** ± 4.1
antmaze-giant-navigate-v0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	**68.4** ± 3.4	50.4 ± 3.7	58.2 ± 4.7	50.4 ± 8.0	**69.2** ± 2.9
antmaze-teleport-navigate-v0	26.4 ± 2.2	35.2 ± 7.5	35.6 ± 5.7	48.4 ± 2.6	49.4 ± 2.7	43.4 ± 6.6	40.8 ± 4.1	**59.8** ± 1.3
humanoidmaze-medium-navigate-v0	6.8 ± 1.3	7.0 ± 2.3	32.6 ± 4.0	82.4 ± 3.6	76.2 ± 3.8	74.8 ± 3.0	52.6 ± 4.3	**89.6** ± 2.9
humanoidmaze-large-navigate-v0	1.0 ± 1.4	1.2 ± 1.6	2.6 ± 1.8	**76.4** ± 4.0	39.0 ± 3.2	23.6 ± 2.9	7.2 ± 3.4	69.4 ± 3.9
humanoidmaze-giant-navigate-v0	0.0 ± 0.0	0.4 ± 0.5	0.0 ± 0.0	**81.4** ± 4.9	34.2 ± 3.3	5.0 ± 1.6	0.0 ± 0.0	66.0 ± 3.4
antsoccer-arena-navigate-v0	5.4 ± 3.0	12.2 ± 4.1	47.4 ± 7.5	39.8 ± 6.5	17.8 ± 2.9	60.6 ± 4.5	56.6 ± 5.5	**76.6** ± 4.0
antsoccer-medium-navigate-v0	1.4 ± 0.5	3.6 ± 2.1	11.6 ± 2.3	**16.6** ± 3.5	2.4 ± 1.5	8.4 ± 1.5	7.2 ± 1.5	15.2 ± 2.4
cube-single-play-v0	5.2 ± 2.2	7.6 ± 2.3	**53.4** ± 5.4	10.0 ± 4.2	1.2 ± 1.1	11.8 ± 2.3	21.8 ± 4.0	41.0 ± 6.1
cube-double-play-v0	1.0 ± 1.0	1.6 ± 1.1	**31.6** ± 3.0	2.2 ± 0.8	0.0 ± 0.0	3.6 ± 2.2	9.4 ± 4.4	**30.8** ± 8.4
scene-play-v0	4.8 ± 2.6	17.2 ± 3.3	44.6 ± 2.9	23.4 ± 7.1	16.6 ± 3.1	37.2 ± 4.2	42.0 ± 5.2	**62.6** ± 5.9
puzzle-3x3-play-v0	3.0 ± 1.2	2.0 ± 1.2	4.2 ± 2.5	4.6 ± 3.2	5.6 ± 1.5	8.8 ± 1.3	**16.0** ± 2.9	13.4 ± 2.3
puzzle-4x4-play-v0	0.0 ± 0.0	0.0 ± 0.0	7.8 ± 2.6	3.4 ± 1.8	13.6 ± 8.4	7.8 ± 2.6	5.4 ± 2.3	**27.6** ± 2.9
Total	167.0	333.6	518.4	920.4	658.0	648.0	623.6	**1090.2**

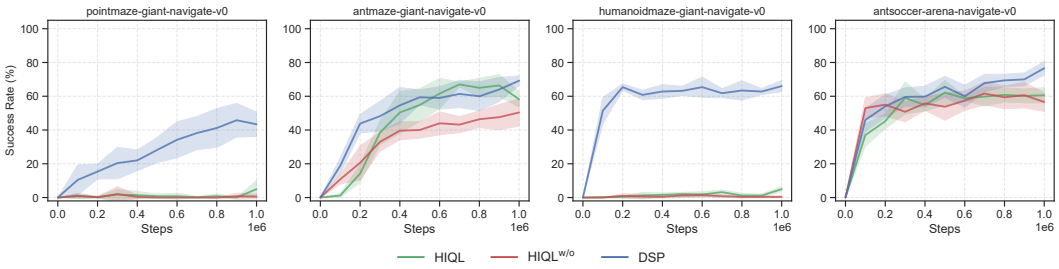


Figure 3: Learning curves on challenging OGBench navigation tasks. DSP generally converges more reliably than HIQL and HIQL$^{w/o}$ in these long-horizon settings, suggesting the benefit of guided generative subgoal planning.

long-horizon planning. In low-dimensional `point` environments, DSP outperforms prior methods in the `medium` and `large` mazes and remains competitive in the more challenging `giant` setting. The learning curves in Figure 3 further show that DSP converges more reliably than HIQL-based baselines in several difficult `giant` environments.

The gains also extend to higher-dimensional robotic domains, including `ant` and `humanoid`. DSP attains near-optimal success rates in `antmaze-medium` and `antmaze-large`, and remains among the strongest methods on several `humanoidmaze` tasks, where high-dimensional states and complex dynamics make long-horizon value-based planning more difficult. These results are consistent with the intended role of DSP: generating data-supported, goal-directed subgoals without relying on explicit value comparisons at the high level.

DSP also achieves the best or near-best performance on the `antsoccer` tasks under both `arena` and `medium` configurations. These environments combine long-range navigation with object interaction, providing a useful test of planning consistency under coupled dynamics. Overall, the locomotion results suggest that guided generative subgoal planning is particularly effective in long-horizon navigation tasks that require stable multi-step subgoal generation.

6.3 Manipulation Results

We further evaluate DSP on the manipulation suite of OGBench, spanning single-object grasping, multi-object interaction, and compositional puzzle solving. Unlike locomotion, these tasks emphasize sequencing local physical interactions rather than long-range navigation.

As shown in Table 1, GCIVL performs strongly on local tasks such as `cube-single` and `cube-double`, where compact goal spaces and short-horizon value estimates provide useful guidance. In these cases, flat policies remain competitive, and hierarchy may offer limited benefit.

DSP shows clearer advantages on multi-step interaction tasks such as `scene-play` and compositional tasks in the `puzzle` suite, where consistent subgoal coordination is more important. By generating data-supported subgoals without explicit high-level value comparisons, DSP performs strongly on `scene-play` and `puzzle-4x4`, while remaining competitive across the suite.

Overall, these results complement the locomotion findings: DSP is most useful when high-level planning requires stable multi-step subgoal generation, while remaining competitive on more local manipulation tasks.

6.4 Guidance, Data Support, and Executor Reachability

To quantify how guidance affects subgoal feasibility, we hold the trained checkpoints and all other settings fixed and vary ω. We measure standardized training-support distance D, state-validity rate V, valid-and-reached rate R, and task success. Lower D indicates greater proximity to the offline data, while higher V and R indicate more valid and executable subgoals. Full metric definitions are provided in Appendix H.

Table 2: Guidance–support–reachability trade-off. D: support distance; V: validity; R: valid-and-reached rate. V, R, and Succ. are percentages.

Datasets	ω	$D\downarrow$	$V\uparrow$	$R\uparrow$	Succ.↑
`AntMaze-Giant`	1	1.6	100.0	68.8	46.0
	3	1.7	100.0	62.1	**69.2**
	5	1.8	98.8	51.6	56.4
	10	2.1	96.9	40.6	27.0
`Scene-Play`	1	1.8	98.0	92.4	35.6
	3	2.0	96.5	90.7	50.4
	5	2.4	94.5	86.6	**62.6**
	10	3.7	84.4	45.3	35.2

Relative to $\omega = 1$, the family-default scales improve success by 23.2 and 27.0 percentage points on Ant-Giant and Scene-Play, respectively, while reducing the valid-and-reached rate by 6.7 and 5.8 points. At $\omega = 10$, support distance increases and validity, reachability, and success all deteriorate. These results show that moderate guidance balances goal direction and executability, whereas excessive guidance is a practical failure mode; DSP does not provide a hard subgoal-feasibility guarantee.

7 Conclusion

In this work, we propose Diffusion Subgoal Planning (DSP), a generative framework for high-level subgoal planning in offline goal-conditioned reinforcement learning. DSP targets a key limitation of hierarchical methods: high-level subgoal selection can become unreliable when guided by noisy goal-conditioned value estimates in long-horizon tasks. Instead of explicitly querying values for high-level planning, DSP learns unconditional and goal-conditioned velocity fields and uses classifier-free guidance to bias generated subgoals toward the desired goal. Combined with a low-level policy trained through implicit value learning, DSP achieves strong performance across challenging OGBench locomotion and manipulation tasks. These results suggest that guided generative subgoal modeling is a promising alternative to explicit value-based high-level planning in long-horizon offline GCRL.

Acknowledgments

This work was supported by the Key Research and Development Program of Jiangsu Province under Grant BE2022095. We thank Qiyu Wang at University College London for helpful discussions and constructive feedback on this work. We also thank the reviewers and Area Chair for their constructive feedback, which helped improve the paper.

References

[1] Hongjoon Ahn, Heewoong Choi, Jisu Han, and Taesup Moon. Option-aware temporally abstracted value for offline goal-conditioned reinforcement learning. *arXiv preprint arXiv:2505.12737*, 2025.

[2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[3] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in Neural Information Processing Systems*, 30, 2017.

[4] Marco Bagatella and Georg Martius. Goal-conditioned offline planning from curious exploration. *Advances in Neural Information Processing Systems*, 36:15358–15383, 2023.

[5] Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev. Goal-conditioned reinforcement learning with imagined subgoals. In *International Conference on Machine Learning*, pages 1430–1440. PMLR, 2021.

[6] Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024.

[7] Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. *arXiv preprint arXiv:2310.07297*, 2023.

[8] Yuanpei Chen, Tianhao Wu, Shengjie Wang, Xidong Feng, Jiechuan Jiang, Zongqing Lu, Stephen McAleer, Hao Dong, Song-Chun Zhu, and Yaodong Yang. Towards human-level bimanual dexterous manipulation with reinforcement learning. *Advances in Neural Information Processing Systems*, 35:5150–5163, 2022.

[9] Jinwoo Choi, Sang-Hyun Lee, and Seung-Woo Seo. Chain-of-goals hierarchical policy for long-horizon offline goal-conditioned RL. *arXiv preprint arXiv:2602.03389*, 2026.

[10] Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 35:35603–35620, 2022.

[11] Meng Fang, Tianyi Zhou, Yali Du, Lei Han, and Zhengyou Zhang. Curriculum-guided hindsight experience replay. *Advances in Neural Information Processing Systems*, 32, 2019.

[12] Zehui Feng, Weichuan Wang, Xiaohan Chen, and Ting Han. PreResQ-R1: Response-preference disentangled ranking-and-scoring reinforcement optimization for robust visual quality assessment. *arXiv preprint arXiv:2511.05393*, 2026.

[13] Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator. *arXiv preprint arXiv:2505.23458*, 2025.

[14] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[15] Divyansh Garg, Skanda Vaidyanath, Kuno Kim, Jiaming Song, and Stefano Ermon. Lisa: learning interpretable skill abstractions from language. *Advances in Neural Information Processing Systems*, 35:21711–21724, 2022.

[16] Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*, 2019.

[17] Vittorio Giammarino, Ruiqi Ni, and Ahmed H Qureshi. Physics-informed value learner for offline goal-conditioned reinforcement learning. *arXiv preprint arXiv:2509.06782*, 2025.

[18] Xudong Gong, Dawei Feng, Kele Xu, Bo Ding, and Huaimin Wang. Goal-conditioned on-policy reinforcement learning. *Advances in Neural Information Processing Systems*, 37:45975–46001, 2024.

[19] Tuomas Haarnoja, Kristian Hartikainen, Pieter Abbeel, and Sergey Levine. Latent space policies for hierarchical reinforcement learning. In *International Conference on Machine Learning*, volume 80, pages 1851–1860. PMLR, 2018.

[20] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[21] Liang He, SiYuan Ma, Qishi Zhan, Yongqi Fan, Mingyu Cao, Qizhen Lan, Zhaolu Kang, and Xilu Wang. Valuspec: Plug-and-play candidate valuation before target verification for tree-based speculative decoding. *arXiv preprint arXiv:2610.31338*, 2026.

[22] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[24] Mineui Hong, Minjae Kang, and Songhwai Oh. Diffused task-agnostic milestone planner. *Advances in Neural Information Processing Systems*, 36:387–405, 2023.

[25] Zixuan Huang, Yating Lin, Fan Yang, and Dmitry Berenson. Subgoal diffuser: Coarse-to-fine subgoal generation to guide model predictive control for robot manipulation. In *IEEE International Conference on Robotics and Automation*, pages 16489–16495, 2024.

[26] Vineet Jain and Siamak Ravanbakhsh. Learning to reach goals via diffusion. In *International Conference on Machine Learning*, pages 21170–21195. PMLR, 2024.

[27] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[28] Kaiqiang Ke, Qian Lin, Zongkai Liu, Shenghong He, and Chao Yu. Conservative offline goal-conditioned implicit V-learning. In *International Conference on Machine Learning*, 2025. Accepted. Available at `https://openreview.net/forum?id=5ryn8tYWHL`.

[29] Donghyeon Ki, JunHyeok Oh, Seong-Woong Shim, and Byung-Jun Lee. Prior-guided diffusion planning for offline reinforcement learning. *arXiv preprint arXiv:2505.10881*, 2025.

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

[31] Sungyoon Kim, Yunseon Choi, Daiki E Matsunaga, and Kee-Eung Kim. Stitching sub-trajectories with conditional diffusion model for goal-conditioned offline rl. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13160–13167, 2024.

[32] Anthony Kobanda, Waris Radji, Mathieu Petitbois, Odalric-Ambrym Maillard, and Rémy Portelas. Offline goal-conditioned reinforcement learning with projective quasimetric planning. *arXiv preprint arXiv:2506.18847*, 2025.

[33] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[34] Seungjae Lee, Jigang Kim, Inkyu Jang, and H Jin Kim. Dhrl: A graph-based approach for long-horizon and sparse hierarchical reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13668–13678, 2022.

[35] Jinning Li, Chen Tang, Masayoshi Tomizuka, and Wei Zhan. Hierarchical planning through goal-conditioned offline reinforcement learning. *IEEE Robotics and Automation Letters*, 7(4): 10216–10223, 2022.

[36] Shuozhe Li, Jincheng Cao, Bodun Hu, Aryan Mokhtari, Leqi Liu, and Amy Zhang. Care-rft: Confidence-anchored reinforcement finetuning for reliable reasoning in large language models. *arXiv preprint arXiv:2602.00085*, 2026.

[37] Shuozhe Li, Vaishnav Tadiparthi, Kwonjoon Lee, Nakul Agarwal, Hossein Nourkhiz Mahjoub, Ehsan Moradi Pari, Lizhang Chen, Amy Zhang, and Liu Leqi. Learning robust reasoning through guided adversarial self-play. *arXiv preprint arXiv:2602.00173*, 2026.

[38] Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *International Conference on Machine Learning*, pages 20035–20064. PMLR, 2023.

[39] Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. Skilldiffuser: interpretable hierarchical planning via skill abstractions in diffusion-based task execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16467–16476, 2024.

[40] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[41] Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Francesco Nex, Hao Cheng, and Michael Ying Yang. Universe: Unified video action models for autonomous driving with flexible mask-modulated modality generation. *arXiv preprint arXiv:2607.05133*, 2026.

[42] Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Michael Ying Yang, Francesco Nex, and Hao Cheng. Driveva: Video action models are zero-shot drivers. In *European Conference on Computer Vision*, pages 315–335, 2026.

[43] Minghuan Liu, Menghui Zhu, and Weinan Zhang. Goal-conditioned reinforcement learning: Problems and solutions. *arXiv preprint arXiv:2201.08299*, 2022.

[44] Ruijia Liu, Ancheng Hou, Shaoyuan Li, and Xiang Yin. Sagas: Semantic-aware graph-assisted stitching for offline temporal logic planning. *arXiv preprint arXiv:2512.00775*, 2025.

[45] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[46] Yaohua Liu, Yifan Guo, and Jiaxin Gao. Anchoring adversarial trajectories to data manifolds: A bilevel transfer optimization framework. *arXiv preprint arXiv:2609.38991*, 2026.

[47] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning*, pages 22825–22855. PMLR, 2023.

[48] Chenrui Ma, Xi Xiao, Tianyang Wang, Xiao Wang, and Yanning Shen. Stochastic interpolants via conditional dependent coupling. *arXiv preprint arXiv:2509.23122*, 2025.

[49] Chenrui Ma, Xi Xiao, Lin Zhao, Tianyang Wang, Ferdinando Fioretto, and Yanning Shen. Drift flow matching. *arXiv preprint arXiv:2605.17244*, 2026.

[50] Guoqing Ma, Yang Chen, Yuhan Zhang, Liangxuan Guo, Yujie Wu, and Shan Yu. Orthogonal space constraint enhances learning scalability and convergence efficiency without gradient backpropagation. *Neurocomputing*, 679:133139, 2026. doi: 10.1016/j.neucom.2026.133139.

[51] Jason Yecheng Ma, Jason Yan, Dinesh Jayaraman, and Osbert Bastani. Offline goal-conditioned reinforcement learning via f-advantage regression. *Advances in Neural Information Processing Systems*, 35:310–323, 2022.

[52] Lina Mezghani, Sainbayar Sukhbaatar, Piotr Bojanowski, Alessandro Lazaric, and Karteek Alahari. Learning goal-conditioned policies offline with self-supervised reward shaping. In *Conference on Robot Learning*, pages 1401–1410. PMLR, 2023.

[53] Evgenii Opryshko, Junwei Quan, Claas Voelcker, Yilun Du, and Igor Gilitschenski. Test-time graph search for goal-conditioned reinforcement learning. *arXiv preprint arXiv:2510.07257*, 2025.

[54] Pengcheng Pan, Guoqing Ma, Yuhan Zhang, Yang Chen, Yichen Liu, Ziheng Li, and Shan Yu. MHWA: Multi-timescale hierarchical world-action model. *arXiv preprint arXiv:2610.14727*, 2026.

[55] Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36:34866–34891, 2023.

[56] Seohong Park, Oleh Rybkin, and Sergey Levine. Metra: Scalable unsupervised rl with metric-aware abstraction. *arXiv preprint arXiv:2310.08887*, 2023.

[57] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*, 2024.

[58] Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. *arXiv preprint arXiv:2506.04168*, 2025.

[59] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025.

[60] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320. PMLR, 2015.

[61] Wenling Shang, Alex Trott, Stephan Zheng, Caiming Xiong, and Richard Socher. Learning world graphs to accelerate hierarchical reinforcement learning. *arXiv preprint arXiv:1907.00664*, 2019.

[62] Wonchul Shin and Yusung Kim. Guide to control: Offline hierarchical reinforcement learning using subgoal generation for long-horizon and sparse-reward tasks. In *International Joint Conference on Artificial Intelligence*, pages 4217–4225, 2023.

[63] Harshit Sikchi, Rohan Chitnis, Ahmed Touati, Alborz Geramifard, Amy Zhang, and Scott Niekum. Smore: score models for offline goal-conditioned reinforcement learning. *arXiv preprint arXiv:2311.02013*, 2023.

[64] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[65] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[66] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[67] Mianchu Wang, Rui Yang, Xi Chen, Hao Sun, Meng Fang, and Giovanni Montana. Goplan: goal-conditioned offline reinforcement learning by planning with learned models. *arXiv preprint arXiv:2310.20025*, 2023.

[68] Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, pages 36411–36430. PMLR, 2023.

[69] Xuesong Wang, Hengrui Zhang, Jiazhi Zhang, C. L. Philip Chen, and Yuhu Cheng. Pcdt: Pessimistic critic decision transformer for offline reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 55(10):7247–7258, 2025. doi: 10.1109/TSMC.2025.3583392.

[70] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[71] Rui Yang, Yiming Lu, Wenzhe Li, Hao Sun, Meng Fang, Yali Du, Xiu Li, Lei Han, and Chongjie Zhang. Rethinking goal-conditioned supervised learning and its connection to offline rl. *arXiv preprint arXiv:2202.04478*, 2022.

[72] Rui Yang, Yong Lin, Xiaoteng Ma, Hao Hu, Chongjie Zhang, and Tong Zhang. What is essential for unseen goal generalization of offline goal-conditioned rl? In *International Conference on Machine Learning*, pages 39543–39571. PMLR, 2023.

[73] Yifu Yuan, Zhenrui Zheng, Zibin Dong, and Jianye Hao. Moduli: Unlocking preference generalization via diffusion models for offline multi-objective reinforcement learning. *arXiv preprint arXiv:2408.15501*, 2024.

[74] Xianghua Zeng, Hao Peng, Angsheng Li, and Yicheng Pan. Structural information-based hierarchical diffusion for offline reinforcement learning. *arXiv preprint arXiv:2509.21942*, 2025.

[75] Tianren Zhang, Shangqi Guo, Tian Tan, Xiaolin Hu, and Feng Chen. Generating adjacency-constrained subgoals in hierarchical reinforcement learning. *Advances in neural information processing systems*, 33:21579–21590, 2020.

A Limitations and Future Work

DSP targets offline, state-based goal-conditioned RL settings where the dataset contains meaningful local reachability structure. As with many offline generative planning methods, performance may be affected when data coverage is limited or evaluation goals deviate substantially from the behavior data. DSP also uses iterative diffusion sampling for high-level subgoal generation, introducing an inference cost controlled by the number of sampling steps; in our experiments, a modest number of steps is sufficient for strong performance.

Future work may extend DSP to partially observable domains, develop more efficient generative parameterizations for faster subgoal sampling, and explore improved learning objectives and optimization strategies for more stable and efficient training [50, 12, 21]. We may also explore multi-timescale hierarchical world-action modeling [54] for subgoal planning, with adversarial training [37] and confidence-aware regularization [36] to improve reliability. Pretrained representations and cross-task transfer, inspired by unified video-action modeling [41, 42], may help reduce dependence on task-specific data coverage.

B Proofs

B.1 Proof of Proposition 4.1

Proof. For simplicity, we assume that T/k is an integer and $k \leq T$. Defining $z_1 := z_{1,T}$ and $z_2 := z_{-1,T}$, the probability of the flat policy π selecting an incorrect action can be computed as follows:

$$\begin{aligned}
\mathcal{E}(\pi) &= \mathbb{P}[\hat{V}(s+1,g) \leq \hat{V}(s-1,g)] && (24)\\
&= \mathbb{P}[\hat{V}(1,T) \leq \hat{V}(-1,T)] && (25)\\
&= \mathbb{P}[-(T-1)(1+\sigma z_1) \leq -(T+1)(1+\sigma z_2)] && (26)\\
&= \mathbb{P}[z_1\sigma(T-1) - z_2\sigma(T+1) \geq 2] && (27)\\
&= \mathbb{P}[z\sigma\sqrt{T^2+1} \leq -\sqrt{2}] && (28)\\
&= \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{T^2+1}}\right), && (29)
\end{aligned}$$

where z is a standard Gaussian random variable, and we use the fact that the sum of two independent Gaussian random variables with standard deviations of σ_1 and σ_2 follows a normal distribution with a standard deviation of $\sqrt{\sigma_1^2+\sigma_2^2}$. Similarly, the probability of the hierarchical policy $\pi^l \circ \pi^h$ selecting an incorrect action is bounded using a union bound as

$$\begin{aligned}
\mathcal{E}(\pi^l \circ \pi^h) &\leq \mathcal{E}(\pi^h) + \mathcal{E}(\pi^l) && (30)\\
&= \mathbb{P}[\hat{V}(s+k,g) \leq \hat{V}(s-k,g)] + \mathbb{P}[\hat{V}(s+1,s+k) \leq \hat{V}(s-1,s+k)] && (31)\\
&= \mathbb{P}[\hat{V}(k,T) \leq \hat{V}(-k,T)] + \mathbb{P}[\hat{V}(1,k) \leq \hat{V}(-1,k)] && (32)\\
&= \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{(T/k)^2+1}}\right) + \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{k^2+1}}\right). && (33)
\end{aligned}$$

□

B.2 Proof of Proposition 4.2

Proof. We first derive the exact one-step decision error for an arbitrary generated subgoal distribution. This general result is then specialized to the matched-distance setting of Proposition 4.2.

Arbitrary generated subgoals. Let $w \sim \pi^h_{\mathrm{DSP}}(\cdot \mid s, g)$ be an arbitrary generated subgoal, assumed to be independent of the low-level value-noise variables. The final goal $g = s + T$ lies to the right of the current state s, so a globally incorrect primitive action moves to the left.

First consider $w > s$ and define $d := w - s > 0$. The distances from the two successor states to w are

$$|s+1-w| = |d-1|, \quad |s-1-w| = d+1. \tag{34}$$

Let $z_+ := z_{s+1,w}, z_- := z_{s-1,w}$, where $z_+, z_- \sim \mathcal{N}(0,1)$ independently. Under the noisy-value model,

$$\hat{V}(s+1,w) = -|d-1|\,(1+\sigma z_+)\,, \tag{35}$$

$$\hat{V}(s-1,w) = -(d+1)\,(1+\sigma z_-)\,. \tag{36}$$

Since both w and the final goal lie to the right of s, the low-level policy makes a globally incorrect decision when it selects the left action:

$$\hat{V}(s+1,w) \le \hat{V}(s-1,w). \tag{37}$$

Substituting the noisy values and rearranging gives

$$\sigma\,(|d-1|z_+ - (d+1)z_-) \ge (d+1) - |d-1|. \tag{38}$$

Using

$$(d+1) - |d-1| = 2\min\{d,1\}, \tag{39}$$

$$|d-1|^2 + (d+1)^2 = 2(d^2+1), \tag{40}$$

we obtain

$$\mathbb{P}\left[\hat{V}(s+1,w) \le \hat{V}(s-1,w) \mid w\right] = \Phi\left(-\frac{\sqrt{2}\min\{d,1\}}{\sigma\sqrt{d^2+1}}\right) =: p_\ell(d). \tag{41}$$

Now consider $w < s$ and define $d := s - w > 0$. By symmetry, $p_\ell(d)$ is the probability that the low-level policy moves to the right, away from the selected subgoal. In this case, however, the right action is globally correct because the final goal lies to the right. Hence, conditioned on $w < s$, the probability of a globally incorrect left action is

$$1 - p_\ell(d).$$

When $w = s$, the two actions are symmetric under the noise model, so the globally incorrect left action is selected with probability $1/2$. Taking expectation over w therefore yields

$$\mathcal{E}\left(\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}\right) = \mathbb{E}_w\left[p_\ell(w-s)\mathbf{1}\{w > s\} + (1 - p_\ell(s-w))\,\mathbf{1}\{w < s\}\right] + \frac{1}{2}\mathbb{P}(w = s). \tag{42}$$

Equation (42) holds for an arbitrary generated subgoal distribution and does not require Gaussianity, symmetry, or unimodality.

Matched-distance DSP error. Under the matched-distance setting of Proposition 4.2, DSP selects $s+k$ with probability $1-\epsilon$ and $s-k$ with probability ϵ. Since $k \ge 1$, Equation (41) gives

$$p_\ell(k) = \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{k^2+1}}\right).$$

If DSP selects $s+k$, the globally incorrect action is chosen with probability $p_\ell(k)$. If DSP selects $s-k$, the globally incorrect action is chosen with probability $1 - p_\ell(k)$. Therefore,

$$\begin{aligned}\mathcal{E}\left(\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}\right) &= (1-\epsilon)p_\ell(k) + \epsilon\,(1 - p_\ell(k)) \\ &= p_\ell(k) + (1 - 2p_\ell(k))\,\epsilon,\end{aligned} \tag{43}$$

which proves Equation (8).

Matched-distance HIQL error. Let H denote the event that HIQL selects the wrong-direction subgoal $s-k$, and let L denote the event that its low-level policy moves away from the selected subgoal. Proposition 4.1 gives

$$\mathbb{P}(H) = p_h(T,k) = \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{(T/k)^2+1}}\right). \tag{44}$$

By symmetry, the conditional low-level error is $p_\ell(k)$ for either selected subgoal:

$$\mathbb{P}(L \mid H) = \mathbb{P}(L \mid H^c) = p_\ell(k). \tag{45}$$

Because the high- and low-level comparisons use independent value-noise variables, H and L are independent.

The final primitive action is globally incorrect in exactly two cases: the high-level direction is correct and the low-level policy moves away from its subgoal, or the high-level direction is incorrect and the low-level policy moves toward its subgoal. Hence,

$$\begin{aligned}\mathcal{E}\left(\pi^\ell \circ \pi^h\right) &= \mathbb{P}(H^c \cap L) + \mathbb{P}(H \cap L^c) \\ &= (1 - p_h(T,k))\, p_\ell(k) + p_h(T,k)\,(1 - p_\ell(k)) \\ &= p_\ell(k) + (1 - 2p_\ell(k))\, p_h(T,k),\end{aligned} \tag{46}$$

which proves Equation (9).

Since $p_\ell(k) < 1/2$, we have

$$\mathcal{E}\left(\pi^\ell_{\mathrm{DSP}} \circ \pi^h_{\mathrm{DSP}}\right) < \mathcal{E}\left(\pi^\ell \circ \pi^h\right) \quad \Longleftrightarrow \quad \epsilon < p_h(T,k), \tag{47}$$

which proves Equation (10).

Gaussian directional-error special case. For the closed-form special case in Equation (11), consider an auxiliary directional proposal

$$u \sim \mathcal{N}\left(s + k, \sigma^2_{\mathrm{data}}\right),$$

where the matched-distance planner selects $s + k$ when $u > s$ and $s - k$ otherwise. This auxiliary proposal is used only to parameterize the wrong-direction probability ϵ. We have

$$\begin{aligned}\epsilon &= \mathbb{P}(u \le s) \\ &= \mathbb{P}\left[\frac{u - (s+k)}{\sigma_{\mathrm{data}}} \le -\frac{k}{\sigma_{\mathrm{data}}}\right] \\ &= \Phi\left(-\frac{k}{\sigma_{\mathrm{data}}}\right).\end{aligned} \tag{48}$$

Condition (10) therefore becomes

$$\Phi\left(-\frac{k}{\sigma_{\mathrm{data}}}\right) < \Phi\left(-\frac{\sqrt{2}}{\sigma\sqrt{(T/k)^2 + 1}}\right). \tag{49}$$

By the strict monotonicity of Φ, this is equivalent to

$$\sqrt{(T/k)^2 + 1} > \frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k}. \tag{50}$$

When

$$\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k} > 1,$$

rearranging Equation (50) yields

$$T > k\sqrt{\left(\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k}\right)^2 - 1}, \tag{51}$$

which is Equation (12). When $\frac{\sqrt{2}\sigma_{\mathrm{data}}}{\sigma k} \le 1$, the left-hand side of Equation (50) is strictly greater than one for every $T > 0$, so no additional positive lower bound on T is required. □

B.3 Proof of Proposition 5.1

Proof. We separate the proof into two parts. We first establish the exact relationship between the conditional density ratio and the behavior-policy goal-reaching value. We then connect classifier-free guidance to Q_β-weighted subgoal extraction.

Density ratio as normalized goal-reaching value. By construction of the future-state relabeling process, $Q_\beta(s, w, g)$ is the conditional probability mass or density of sampling g from the future continuation after first selecting w:

$$p_{\mathcal{D}}(g \mid s, w) = Q_\beta(s, w, g). \tag{52}$$

Moreover, $p_{\mathcal{D}}(w \mid s) = \pi_\beta(w \mid s)$. Marginalizing over the behavior subgoal distribution gives

$$p_{\mathcal{D}}(g \mid s) = \int p_{\mathcal{D}}(g \mid s, w) p_{\mathcal{D}}(w \mid s)\, dw = \int Q_\beta(s, w, g) \pi_\beta(w \mid s)\, dw = V_\beta(s, g). \tag{53}$$

Bayes' rule therefore yields

$$p_{\mathcal{D}}(w \mid s, g) = \frac{p_{\mathcal{D}}(g \mid s, w) p_{\mathcal{D}}(w \mid s)}{p_{\mathcal{D}}(g \mid s)} = \pi_\beta(w \mid s) \frac{Q_\beta(s, w, g)}{V_\beta(s, g)}. \tag{54}$$

Dividing both sides by $p_{\mathcal{D}}(w \mid s) = \pi_\beta(w \mid s)$ gives

$$\frac{p_{\mathcal{D}}(w \mid s, g)}{p_{\mathcal{D}}(w \mid s)} = \frac{Q_\beta(s, w, g)}{V_\beta(s, g)}. \tag{55}$$

Using

$$Q_\beta(s, w, g) = V_\beta(s, g) + A_\beta(s, w, g), \tag{56}$$

we obtain

$$\frac{p_{\mathcal{D}}(w \mid s, g)}{p_{\mathcal{D}}(w \mid s)} = 1 + \frac{A_\beta(s, w, g)}{V_\beta(s, g)}. \tag{57}$$

Taking logarithms yields

$$\widetilde{A}_\beta(s, w, g) = \log \frac{p_{\mathcal{D}}(w \mid s, g)}{p_{\mathcal{D}}(w \mid s)} = \log \frac{Q_\beta(s, w, g)}{V_\beta(s, g)} = \log \left(1 + \frac{A_\beta(s, w, g)}{V_\beta(s, g)} \right). \tag{58}$$

For fixed (s, g), $V_\beta(s, g)$ does not depend on w. Wherever $Q_\beta(s, w, g) > 0$,

$$\begin{aligned} \frac{\partial \widetilde{A}_\beta(s, w, g)}{\partial A_\beta(s, w, g)} &= \frac{1}{V_\beta(s, g) + A_\beta(s, w, g)} \\ &= \frac{1}{Q_\beta(s, w, g)} > 0. \end{aligned} \tag{59}$$

Thus, $\widetilde{A}_\beta$ is a strictly increasing transformation of the behavior-policy RL advantage. In particular,

$$\widetilde{A}_\beta(s, w, g) > 0 \quad \Longleftrightarrow \quad A_\beta(s, w, g) > 0, \tag{60}$$

and both quantities induce the same ordering over w for fixed (s, g).

Classifier-free guidance as Q_β-weighted extraction. We next connect the density-ratio identity to the guided velocity field used by DSP. Under the score-approximation interpretation adopted in Proposition 5.1, the unconditional and goal-conditioned velocity fields satisfy

$$v_\theta(w^i, i, s, \emptyset) \approx \nabla_{w^i} \log p_i(w^i \mid s), \quad v_\theta(w^i, i, s, g) \approx \nabla_{w^i} \log p_i(w^i \mid s, g), \tag{61}$$

where p_i denotes the corresponding distribution at diffusion time i. Therefore, their difference satisfies

$$v_\theta(w^i, i, s, g) - v_\theta(w^i, i, s, \emptyset) \approx \nabla_{w^i} \log \frac{p_i(w^i \mid s, g)}{p_i(w^i \mid s)}. \tag{62}$$

Substituting these approximations into the CFG velocity field gives

$$\begin{aligned} \hat{v}_\theta(w^i, i) &= v_\theta(w^i, i, s, \emptyset) + \omega \left(v_\theta(w^i, i, s, g) - v_\theta(w^i, i, s, \emptyset) \right) \\ &\approx \nabla_{w^i} \left[\log p_i(w^i \mid s) + \omega \log \frac{p_i(w^i \mid s, g)}{p_i(w^i \mid s)} \right] \\ &= \nabla_{w^i} \log \left[p_i(w^i \mid s) \left(\frac{p_i(w^i \mid s, g)}{p_i(w^i \mid s)} \right)^\omega \right]. \end{aligned} \tag{63}$$

Under the corresponding endpoint-density interpretation of CFG, the induced subgoal distribution is

$$p_\omega(w \mid s, g) \propto p(w \mid s) \left(\frac{p(w \mid s, g)}{p(w \mid s)} \right)^\omega. \tag{64}$$

Using $p(w \mid s) = \pi_\beta(w \mid s)$ and the density-ratio identity

$$\frac{p(w \mid s, g)}{p(w \mid s)} = \frac{Q_\beta(s, w, g)}{V_\beta(s, g)}, \tag{65}$$

we obtain

$$p_\omega(w \mid s, g) \propto \pi_\beta(w \mid s) \left(\frac{Q_\beta(s, w, g)}{V_\beta(s, g)} \right)^\omega \propto \pi_\beta(w \mid s) Q_\beta(s, w, g)^\omega, \tag{66}$$

where $V_\beta(s, g)^\omega$ is absorbed into the normalization constant because it does not depend on w. Equivalently,

$$p_\omega(w \mid s, g) \propto \pi_\beta(w \mid s) \exp\left(\omega \widetilde{A}_\beta(s, w, g) \right), \tag{67}$$

where $\widetilde{A}_\beta(s, w, g) = \log \frac{Q_\beta(s,w,g)}{V_\beta(s,g)}$. □

C Implementation Details

C.1 Goal Mixed Sampling

We adopt the standard HER [3, 57] trick provided by OGBench. To train both value networks and actors, we employ a mixture of three distinct goal distributions:

- $p_{\mathrm{cur}}(g \,|\, s)$: A degenerate distribution at the current state s (i.e., $g = s$).
- $p_{\mathrm{traj}}(g \,|\, s)$: A uniform distribution over future states visited within the same trajectory as the current state s.
- $p_{\mathrm{rand}}(g \,|\, s)$: A uniform distribution over arbitrary states sampled from the entire dataset, independent of the current state s.

The specific mixing ratios for each task are detailed in Table 3.

C.2 Target Value Network

In our proposed method, we retain the conventional use of target value networks to further stabilize the training process. Previous studies [60, 5] have demonstrated that this technique facilitates faster convergence with negligible computational overhead.

C.3 Architectures

In our experiments, the value networks and policies across all methods are implemented as three-layer MLPs that take the concatenation of states and goals as input.

For most hierarchical baselines, an additional subgoal encoder is used to map subgoals into a 10-dimensional latent space, which serves as the interface between the high-level planner and the low-level policy. In contrast, DSP directly operates in the original subgoal space and therefore does not introduce an explicit subgoal encoder. We also report results for HIQL w/o rep, which removes this subgoal encoding for comparison.

Empirical results for HIQL w/o rep in Table 1 indicate that subgoal encoding can noticeably improve performance in high-dimensional state spaces, with the effect being particularly evident in challenging environments such as `humanoidmaze`. Although computational constraints prevent a systematic ablation of architectural choices, this observation is consistent with prior work on representation learning [68, 10, 56, 32], which suggests that appropriate dimensional compression can facilitate learning and planning in complex environments.

D Hyperparameters

To support reproducibility, Table 3 reports the shared hyperparameters, while Table 4 lists environment-specific settings for the guidance scale ω, policy temperature α, and subgoal interval k. We select ω via a coarse sweep on representative tasks and then fix it across all tasks and seeds within each environment family; k is likewise shared within each family. The number of flow integration steps is fixed globally at $N = 20$.

Table 3: Common hyperparameters for DSP in our experiments.

Hyperparameter	**Value**
Learning rate	0.0003
Optimizer	Adam
Batch size	1024
Total gradient steps	1000000
MLP dimensions	$[512, 512, 512]$
Activation function	GELU
Target network smoothing coefficient κ	0.005
Discount factor γ	0.995 (humanoidmaze), 0.99 (others)
Expectile ε	0.7
Diffusion time sampling distribution	$\mathcal{U}(0, 1)$
Diffusion step N	20
Dropout probability p_{unc}	0.1
Value goal mix ratio $(p_{\text{cur}}, p_{\text{traj}}, p_{\text{rand}})$	$(0.2, 0.5, 0.3)$
Policy goal mix ratio $(p_{\text{cur}}, p_{\text{traj}}, p_{\text{rand}})$	$(0, 1.0, 0)$

Table 4: Task specific hyperparameters for DSP in our experiments.

Environment	**Datasets**	ω	α	k
`pointmaze`	`pointmaze-medium-navigate-v0`	3.0	3.0	25
	`pointmaze-large-navigate-v0`	3.0	3.0	25
	`pointmaze-giant-navigate-v0`	3.0	3.0	25
	`pointmaze-teleport-navigate-v0`	3.0	3.0	25
	`pointmaze-medium-stitch-v0`	3.0	3.0	25
	`pointmaze-large-stitch-v0`	3.0	3.0	25
	`pointmaze-giant-stitch-v0`	3.0	3.0	25
	`pointmaze-teleport-stitch-v0`	3.0	3.0	25
`antmaze`	`antmaze-medium-navigate-v0`	3.0	3.0	25
	`antmaze-large-navigate-v0`	3.0	3.0	25
	`antmaze-giant-navigate-v0`	3.0	3.0	25
	`antmaze-teleport-navigate-v0`	3.0	3.0	25
	`antmaze-medium-stitch-v0`	3.0	3.0	25
	`antmaze-large-stitch-v0`	3.0	3.0	25
	`antmaze-giant-stitch-v0`	3.0	3.0	25
	`antmaze-teleport-stitch-v0`	3.0	3.0	25
`humanoidmaze`	`humanoidmaze-medium-navigate-v0`	1.5	3.0	100
	`humanoidmaze-large-navigate-v0`	1.5	3.0	100
	`humanoidmaze-giant-navigate-v0`	1.5	3.0	100
	`humanoidmaze-medium-stitch-v0`	1.5	3.0	100
	`humanoidmaze-large-stitch-v0`	1.5	3.0	100
	`humanoidmaze-giant-stitch-v0`	1.5	3.0	100
`antsoccer`	`antsoccer-arena-navigate-v0`	3.0	3.0	25
	`antsoccer-medium-navigate-v0`	3.0	3.0	25
	`antsoccer-arena-stitch-v0`	3.0	3.0	25
	`antsoccer-medium-stitch-v0`	3.0	3.0	25
`cube`	`cube-single-play-v0`	5.0	3.0	20
	`cube-double-play-v0`	5.0	3.0	20
`scene`	`scene-play-v0`	5.0	3.0	20
`puzzle`	`puzzle-3x3-play-v0`	5.0	3.0	20
	`puzzle-4x4-play-v0`	5.0	3.0	20
`D4RL-antmaze`	`antmaze-umaze-v2`	3.0	3.0	25
	`antmaze-umaze-diverse-v2`	3.0	3.0	25
	`antmaze-medium-play-v2`	3.0	3.0	25
	`antmaze-medium-diverse-v2`	3.0	3.0	25
	`antmaze-large-play-v2`	3.0	3.0	25
	`antmaze-large-diverse-v2`	3.0	3.0	25

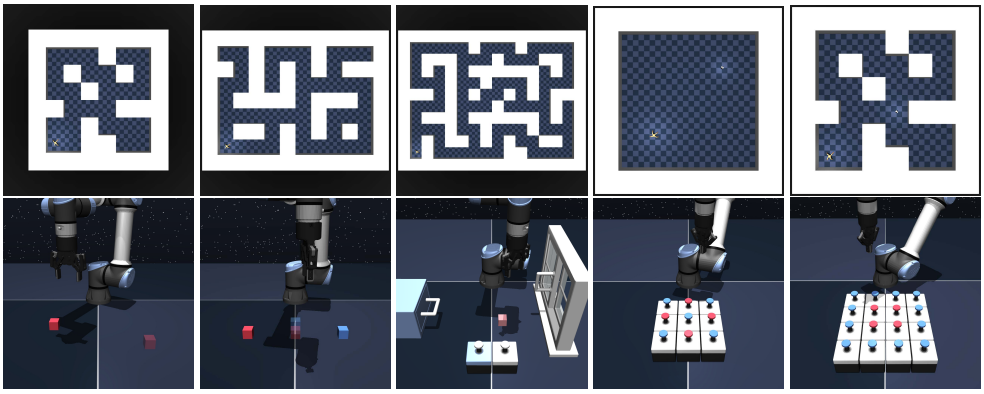


Figure 4: Illustration of the OGBench environments, covering diverse locomotion and manipulation tasks used for evaluation.

E Computational Resources

All experiments were conducted on two compute servers, each equipped with four NVIDIA GeForce RTX 4090 GPUs (24 GB of VRAM per GPU). Unless otherwise specified, all reported training times correspond to runs executed on a single GPU. For locomotion tasks, training DSP on the largest environment (`humanoidmaze-giant-navigate-v0`) requires approximately one hour on a single GPU. For manipulation tasks, the corresponding training time is around 40 minutes.

F Environments and Datasets

This section provides a detailed description of each task. Visualizations of the environments are presented in Figure 4. For further technical specifications, we refer readers to the OGBench white paper.

Maze. `Maze` is a challenging sparse-reward, long-horizon locomotion task requiring the agent to navigate from an arbitrary starting position to a specified goal. The environments vary by agent complexity: the 2-DoF `point`, the 8-DoF `ant`, and the 21-DoF `humanoid`. Each maze is categorized by size: `medium`, `large`, and `giant`. Additionally, the `teleport` configuration mirrors the large layout but incorporates random teleporters—some leading to dead ends—to test robustness against environmental stochasticity. The datasets are distinct based on collection methods; specifically, the navigate datasets consist of full trajectories collected by a noisy expert policy attempting to reach randomly sampled goals. In contrast, the `stitch` datasets are composed of short goal-reaching trajectories with a maximum length of four cell units, requiring the agent to stitch together multiple trajectory segments to solve long-horizon tasks.

AntSoccer. `Antsoccer` is introduced to further evaluate multi-goal generalization. In this environment, a quadrupedal Ant robot must dribble a ball to a target location. This task exceeds the difficulty of `antmaze`, as it requires simultaneous navigation and precise ball manipulation. The environment includes two layouts: `arena`, an open space without walls, and `medium`, which corresponds to the layout in `antmaze`.

Cube. `Cube` involves complex robotic manipulation focused on pick-and-place operations, where a robotic arm must arrange cubes into a specific configuration. The training dataset consists of `play`-style data generated by a scripted policy that randomly moves and stacks cubes. During evaluation, the agent is tasked with moving, stacking, swapping, or arranging cubes to match a goal configuration. Success requires learning generalizable multi-object behaviors and long-horizon reasoning from unstructured, stochastic trajectories.

Scene. The `scene` task evaluates sequential reasoning and long-horizon planning involving everyday objects such as cubes, windows, drawers, and button locks. Pressing a button toggles the locking state of a corresponding object (e.g., a drawer or window). Similar to `cube`, the dataset is collected via a `play`-style scripted policy. At test time, the agent must complete a sequence of sub-tasks to achieve the final goal, necessitating the ability to compose skills temporally.

Puzzle. `Puzzle` requires a robotic arm to solve a "Lights Out" style problem. The workspace contains a 2D array of buttons (e.g., a 3x3 grid); pressing a button toggles its color and that of its orthogonal neighbors. The objective is to achieve a specific color configuration through a sequence of presses. This task demands both precise low-level continuous control and high-level combinatorial generalization, posing a significant challenge to the agent's long-term reasoning capabilities.

G Offline GCRL Baseline Algorithms

A brief overview of the baseline algorithms compared in this study is provided below:

Goal-conditioned behavioral cloning (GCBC). [16] GCBC is a standard imitation learning approach that clones behaviors by utilizing hindsight goal relabeling with future states observed within the same trajectory.

Classifier-free guidance reinforcement learning (CFGRL). [13] CFGRL employs a diffusion model as the policy network and executes action inference through classifier-free guidance sampling, enhancing the generation of high-fidelity actions.

Goal-conditioned implicit V learning (GCIVL). [57] GCIVL introduces a V-only objective to regress the optimal state-value function. By omitting Q-value learning, it does not explicitly marginalize over non-causal factors, which can introduce optimistic bias in stochastic environments.

Option-aware temporally abstracted V learning (OTA). [1] OTA incorporates an auxiliary high-level value function to facilitate high-level policy extraction. Its core contribution involves leveraging option-aware temporal abstraction to derive this value function, thereby providing more precise guidance for the high-level policy.

Physics-informed value learner (Pi-HIQL). [17] Drawing inspiration from the Eikonal equation, Pi-HIQL incorporates an Eikonal regularizer into implicit V-learning to encourage geometric structure in the learned value function. This method aims to induce geometric inductive biases within the value function by leveraging ground-truth physical constraints.

Hierarchical implicit Q learning (HIQL). [55] The theoretical framework of HIQL is detailed in Section 3. Regarding implementation, standard HIQL employs a latent subgoal representation for policy extraction. Specifically, the high-level and low-level policies are parameterized as $\pi^h : \mathcal{S} \times \mathcal{S} \to \Delta(\mathcal{Z})$ and $\pi^\ell : \mathcal{S} \times \mathcal{Z} \to \Delta(\mathcal{A})$, respectively, where $\mathcal{Z}$ denotes the latent subgoal space.

HIQL without subgoal representation (HIQL$^{\mathrm{w/o}}$). This variant removes the latent subgoal representation mechanism used in the standard HIQL implementation. Instead, it adheres strictly to the objectives defined in Equation 3 and Equation 4 of Section 3 for policy learning, operating without the auxiliary latent mapping.

H Subgoal Support and Executor Reachability

Neither DSP nor HIQL provides a hard feasibility guarantee for generated subgoals. To quantify how classifier-free guidance affects data support and low-level execution, we hold the trained checkpoints and all other settings fixed and vary the guidance scale ω.

For M generated subgoals $\{w_i\}_{i=1}^M$, we define the standardized training-support distance as

$$D = \frac{1}{M} \sum_{i=1}^{M} \min_{x \in \mathcal{S}_\mathcal{D}} \left\| \frac{w_i - x}{\sigma_\mathcal{D}} \right\|_2, \tag{68}$$

where $\mathcal{S}_\mathcal{D}$ denotes the states contained in the offline dataset and $\sigma_\mathcal{D}$ is their coordinate-wise standard deviation. A smaller D indicates that generated subgoals remain closer to the training support.

We further measure the state-validity rate

$$V = \frac{1}{M} \sum_{i=1}^{M} \mathbf{1}\left[\mathrm{Valid}(w_i)\right], \tag{69}$$

and the valid-and-reached rate

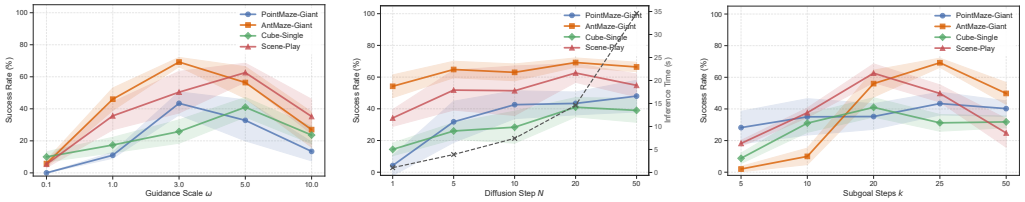
$$R = \frac{1}{M} \sum_{i=1}^{M} \mathbf{1}\left[\text{Valid}(w_i)\right] \mathbf{1}\left[d(s_{i,k}, w_i) \leq \delta\right], \tag{70}$$

where $s_{i,k}$ is the state reached after executing the low-level policy toward w_i for at most k steps. Both V and R are reported as percentages.

As shown in Table 2, the family-default guidance scales improve task success relative to $\omega = 1$, while moderately reducing the valid-and-reached rate. With excessive guidance ($\omega = 10$), support distance increases and validity, reachability, and success all deteriorate. Thus, moderate guidance balances goal direction and executability, whereas excessive guidance constitutes a practical failure mode.

I Hyperparameter Sensitivity

I.1 Guidance Scale



(a) Guidance Scale ω (b) Diffusion Step N (c) Subgoal Steps k

Figure 5: Hyperparameter sensitivity analysis of DSP. (a): sensitivity to the guidance scale ω under classifier-free guidance, showing a consistent inverted-U trend across representative tasks. (b): sensitivity to the number of diffusion steps N, illustrating the trade-off between performance gains and inference cost. (c): sensitivity to the subgoal steps k, showing a broad performance plateau with degradation for overly large k. Shaded regions indicate one standard deviation over 5 random seeds.

Figure 5(a) presents a sensitivity analysis of the guidance scale ω across four representative tasks. The results exhibit a consistent inverted-U trend, from which we draw three key observations. First, increasing ω from the baseline value of 1.0 (which approximates standard conditional sampling) to the range of 3.0 to 5.0 yields substantial performance gains across all tasks. For instance, in `pointmaze-giant-navigate-v0`, the success rate increases from 11.0% to 43.4%. This is consistent with the implicit advantage-weighted interpretation of CFG in DSP, where stronger guidance increases preference for goal-relevant subgoals. Second, we observe clear task-dependent preferences for the guidance strength. Locomotion tasks tend to peak around $\omega = 3.0$, whereas manipulation tasks benefit from stronger guidance and achieve their best performance at $\omega = 5.0$. We hypothesize that manipulation tasks involve complex contact dynamics and narrow optimal regions, requiring stronger directional signals to steer generation toward effective subgoals. In contrast, navigation tasks span larger spatial scales, where overly strong guidance may overly constrain subgoal diversity and hinder exploration of feasible paths. Finally, performance often degrades when ω becomes excessively large. This behavior aligns with prior findings on CFG [22]: excessive guidance pushes samples toward low-density regions of the data distribution, increasing the likelihood of generating unreachable subgoals and ultimately impairing low-level execution.

I.2 Diffusion Step

We further study the effect of the number of diffusion steps N on both the performance and inference efficiency of DSP. Figure 5(b) reports the average success rates and standard deviations for $N \in \{1, 5, 10, 20, 50\}$. Across all tasks, DSP exhibits a clear pattern in which performance improves substantially as N increases, followed by saturation and occasional mild degradation. Increasing the number of diffusion steps from $N = 1$ to $N = 10$ or $N = 20$ consistently leads to performance gains across all tasks, indicating that additional diffusion steps allow the conditional generative model to better capture the structural information required for long-horizon subgoal generation. This effect is particularly pronounced on more challenging tasks such as `pointmaze-giant` and `scene-play`.

In contrast, further increasing the number of diffusion steps to $N = 50$ does not yield consistent improvements, and even results in slight performance drops on some tasks (e.g., `antmaze-giant` and `scene-play`), suggesting diminishing returns and potential instability introduced by excessive sampling.

At the same time, inference time increases approximately linearly with the number of diffusion steps, rising from about 1 second at $N = 1$ to over 35 seconds at $N = 50$, leading to a substantial computational overhead. Here, the reported inference time is measured during evaluation over 10000 repeated inferences, where each inference generates one subgoal and one action. Considering the trade-off between performance and efficiency, $N = 20$ achieves near-optimal or optimal performance on most tasks while maintaining a reasonable inference cost. We therefore adopt $N = 20$ as the default number of diffusion steps in all experiments unless otherwise specified.

I.3 Subgoal Steps

We study the sensitivity of DSP to the subgoal steps k, which controls the temporal distance of high-level subgoals. As shown in Figure 5(c), performance consistently improves as k increases from very small values, indicating that short-horizon subgoals are insufficient for effective long-range planning. As k increases further, performance reaches a broad plateau (e.g., $k = 20$–25), suggesting that DSP is not highly sensitive to the exact choice of k once it exceeds a reasonable scale.

However, when k becomes overly large, performance may degrade. This is because distant subgoals are harder for the low-level policy to reliably reach, and such long-range transitions are less supported in the offline dataset, leading to increased uncertainty in subgoal generation.

In practice, we adopt a simple and consistent heuristic across domains: larger k for long-horizon navigation tasks and smaller k for manipulation tasks requiring fine-grained control. This choice is consistent with common practices in hierarchical reinforcement learning and does not require per-task tuning.

J High-Level Subgoal Generation: Ablations and Comparisons

J.1 Controlled Comparison of High-Level Mechanisms

In DSP, training subgoals are constructed from k-step future states in the offline dataset and are not assumed to be optimal. CFG therefore does not provide optimal labels during training; instead, it introduces an inference-time goal-directed bias in the learned generative subgoal policy. To isolate this effect, we conduct a controlled ablation on the high-level planner while keeping the low-level policy training objective fixed as HIQL-style AWR.

Table 5: Ablation on high-level subgoal generation. The table reports average binary success rate (%) over 5 seeds. MLE and diffusion variants are trained on the same k-step future-state subgoal targets.

Datasets	**MLE**	**Diffusion** ($\omega = 1$)	**HIQL**	**DSP**
`antmaze-medium-navigate-v0`	75.8 ± 2.5	77.0 ± 7.5	93.2 ± 1.3	$\mathbf{98.4} \pm 0.5$
`antmaze-large-navigate-v0`	58.0 ± 5.1	58.2 ± 7.6	87.8 ± 1.5	$\mathbf{92.0} \pm 4.1$
`antmaze-giant-navigate-v0`	14.8 ± 4.4	43.8 ± 4.0	58.2 ± 4.7	$\mathbf{69.2} \pm 2.9$
`cube-single-play-v0`	15.0 ± 3.7	19.2 ± 3.8	11.8 ± 2.3	$\mathbf{41.0} \pm 6.1$
`cube-double-play-v0`	1.2 ± 1.1	4.8 ± 0.8	3.6 ± 2.2	$\mathbf{30.8} \pm 8.4$
`scene-play-v0`	6.8 ± 1.8	25.6 ± 4.2	37.2 ± 4.2	$\mathbf{62.6} \pm 5.9$

We compare four high-level planning mechanisms: (i) an MLE-based subgoal generator trained to imitate k-step future states, (ii) diffusion-based conditional sampling with $\omega = 1$, (iii) HIQL-style value-based high-level planning, and (iv) DSP with CFG-guided subgoal generation. All methods use the same type of low-level AWR executor. The results are reported in Table 5.

The MLE-based subgoal generator performs substantially worse than HIQL and DSP, especially on long-horizon navigation tasks, indicating that directly imitating k-step future states is insufficient. Diffusion-based conditional sampling improves over MLE in several settings, suggesting that modeling a richer subgoal distribution is beneficial. However, the gap between diffusion with $\omega = 1$ and DSP shows that standard conditional generation does not explain the full performance

gains. CFG-guided sampling further biases generation toward goal-relevant subgoals and achieves the strongest performance across all evaluated tasks.

These results suggest that DSP is not merely a k-step subgoal imitation method. Rather, its performance comes from combining data-supported generative subgoal modeling with inference-time goal-directed guidance, consistent with the advantage-like interpretation of CFG discussed in Section 5.3.

J.2 Comparison with Diffusion-Based Hierarchical Planners

Table 6: Success rates (%) under the matched OGBench protocol, reported as mean ± standard deviation over five seeds. Bold indicates the highest mean in each setting.

Datasets	Representation	Merlin	HDMI	HD	DSP
`PointMaze-Large`	*Full* = XY	43.2 ± 8.1	61.2 ± 4.1	73.6 ± 6.8	$\mathbf{93.6} \pm 2.9$
`PointMaze-Giant`	*Full* = *XY*	0.4 ± 0.5	26.4 ± 5.4	12.6 ± 6.9	$\mathbf{43.4} \pm 7.2$
`AntMaze-Large`	*Full*	22.6 ± 5.2	53.4 ± 5.6	64.0 ± 8.2	$\mathbf{92.0} \pm 4.1$
	XY	34.2 ± 5.8	65.0 ± 3.7	74.4 ± 5.5	$\mathbf{96.0} \pm 2.0$
`AntMaze-Giant`	*Full*	0.0 ± 0.0	0.0 ± 0.0	0.2 ± 0.4	$\mathbf{69.2} \pm 2.9$
	XY	0.0 ± 0.0	22.8 ± 3.5	8.6 ± 3.7	$\mathbf{77.2} \pm 2.7$

We compare DSP with Merlin [26], HDMI [38], and HD [6]. All methods use the same OGBench datasets, 1M training updates, five seeds, and the same five-goal evaluation protocol. We reimplement the three baselines in JAX while retaining their method-specific architectures and tuning their key hyperparameters.

Because these methods originally use different goal representations, we evaluate two interfaces on AntMaze: *Full* uses full OGBench goal observations and full-state subgoals, while *XY* uses only two-dimensional positions for both goals and subgoals. PointMaze states are already two-dimensional, so *Full* and *XY* are equivalent.

As shown in Table 6, DSP achieves the highest mean success in all six evaluated task–representation settings. HDMI and HD generally improve under the lower-dimensional *XY* interface, confirming that goal representation affects high-level modeling difficulty. DSP remains strongest under both *Full* and *XY* representations in this matched evaluation protocol.

K Inference-Time Analysis

We analyze the inference-time efficiency of DSP and compare it with hierarchical baselines under matched budgets. In addition to the performance–latency trade-off of DSP shown in Appendix I.2, we provide a direct comparison across methods.

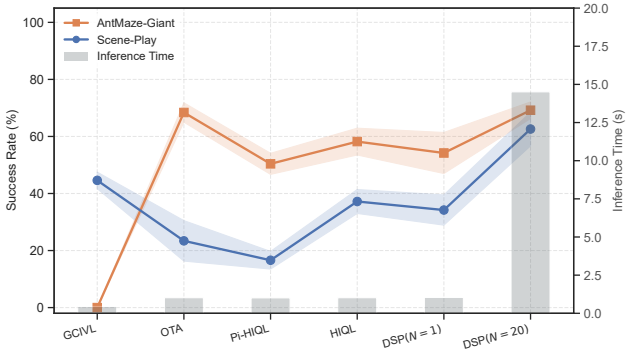


Figure 6: Comparison of success rate and inference time across methods. The left axis shows success rates on `antmaze-giant` and `scene-play`, while the right axis shows inference time. DSP with $N = 1$ corresponds to the closest matched-budget setting, while increasing the number of diffusion steps (e.g., $N = 20$) leads to improved performance at higher inference cost.

Table 7: Training time comparison of different methods measured on the same hardware.

Method	**GCIVL**	**OTA**	**Pi-HIQL**	**HIQL**	**DSP**
Training Time (min)	24	56	53	42	57

As shown in Fig. 6, DSP with $N = 1$ achieves performance comparable to HIQL under similar inference time, indicating that DSP does not rely on increased computation to be effective. Increasing the number of diffusion steps further improves performance (e.g., $+11$ on `antmaze-giant` and $+25$ on `scene-play` for $N = 20$), demonstrating a favorable performance–latency trade-off.

We note that inference time is measured during evaluation over 10,000 repeated inferences, where each inference consists of generating one subgoal and one action. We also report the training time of different methods on the same hardware for reference, as shown in Table 7.

L Stitching Experiments

We evaluate the stitching capability of DSP on the OGBench stitch datasets, which test whether an algorithm can solve long-horizon tasks by composing shorter offline trajectory fragments [57, 69]. Table 8 compares DSP with flat and hierarchical offline GCRL baselines. We include several DSP variants to isolate the effects of goal sampling and guidance strength. The superscript "mix" denotes the variant trained with the OGBench actor goal mixing strategy for stitching tasks, i.e., $(p_{\text{cur}}, p_{\text{traj}}, p_{\text{rand}}) = (0, 0.5, 0.5)$. The subscript 1 denotes standard conditional sampling with guidance scale $\omega = 1$. DSP without a subscript uses the task-specific guidance scale reported in Appendix D.

Table 8: Complete comparison between DSP and the offline GCRL baselines. The table reports the average binary success rate (%) across five test-time goals for each task, averaged over 5 seeds. Standard deviations are indicated by the $\pm$ symbol. Entries within 95% of the best-performing value in each row are highlighted in **bold**.

Datasets	**Flat Policies**			**Hierarchical Policies**					
	GCBC	**CFGRL**	**GCIVL**	**HIQL**	**HIQL**$^{\text{w/o}}$	**DSP**$_1^{\text{mix}}$	**DSP**$_1$	**DSP**$^{\text{mix}}$	**DSP**
`pointmaze-medium-stitch-v0`	23.2 ± 16.7	24.6 ± 8.3	20.8 ± 4.0	62.0 ± 6.8	$\mathbf{69.4} \pm 13.6$	2.4 ± 0.9	54.6 ± 3.0	5.0 ± 2.0	65.8 ± 5.8
`pointmaze-large-stitch-v0`	4.4 ± 8.3	13.0 ± 3.9	8.0 ± 11.0	22.0 ± 7.5	8.2 ± 7.7	0.0 ± 0.0	38.0 ± 1.9	3.6 ± 1.1	$\mathbf{44.2} \pm 4.3$
`pointmaze-giant-stitch-v0`	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{1.2} \pm 0.8$
`pointmaze-teleport-stitch-v0`	26.8 ± 8.7	13.8 ± 8.9	28.6 ± 5.0	33.2 ± 2.5	8.4 ± 4.9	19.6 ± 4.2	43.4 ± 7.5	14.0 ± 5.7	$\mathbf{51.8} \pm 4.9$
`antmaze-medium-stitch-v0`	52.6 ± 5.7	44.4 ± 5.3	32.6 ± 10.4	$\mathbf{89.8} \pm 3.8$	$\mathbf{89.6} \pm 3.4$	66.6 ± 9.2	75.8 ± 6.6	65.8 ± 5.5	$\mathbf{85.8} \pm 3.3$
`antmaze-large-stitch-v0`	1.8 ± 2.5	4.4 ± 4.7	5.0 ± 4.9	$\mathbf{63.6} \pm 8.6$	$\mathbf{66.4} \pm 3.8$	19.6 ± 5.3	22.0 ± 6.9	48.2 ± 2.9	51.8 ± 4.9
`antmaze-giant-stitch-v0`	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.8 ± 0.8	1.8 ± 2.0	2.6 ± 1.3	$\mathbf{9.0} \pm 4.0$	$\mathbf{8.6} \pm 2.6$
`antmaze-teleport-stitch-v0`	31.4 ± 7.7	13.0 ± 6.0	27.2 ± 5.4	34.0 ± 3.2	33.8 ± 3.4	26.8 ± 6.6	$\mathbf{45.2} \pm 6.0$	26.8 ± 4.7	39.2 ± 6.1
`humanoidmaze-medium-stitch-v0`	26.2 ± 5.4	18.6 ± 6.8	31.8 ± 6.1	$\mathbf{73.8} \pm 3.9$	62.4 ± 3.0	53.2 ± 3.3	58.6 ± 4.0	50.8 ± 6.3	$\mathbf{73.2} \pm 4.3$
`humanoidmaze-large-stitch-v0`	4.2 ± 1.7	4.0 ± 2.1	5.0 ± 3.8	$\mathbf{19.2} \pm 2.9$	8.2 ± 1.9	2.4 ± 1.3	4.0 ± 2.1	8.2 ± 3.6	$\mathbf{19.8} \pm 4.4$
`humanoidmaze-giant-stitch-v0`	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	1.8 ± 1.9	0.0 ± 0.0	0.2 ± 0.4	0.0 ± 0.0	17.6 ± 7.0	$\mathbf{35.2} \pm 5.9$
`antsoccer-arena-stitch-v0`	21.6 ± 3.9	12.6 ± 6.4	3.8 ± 1.8	14.2 ± 2.8	35.6 ± 3.4	15.6 ± 3.8	16.6 ± 4.7	24.4 ± 5.7	$\mathbf{40.2} \pm 5.2$
`antsoccer-medium-stitch-v0`	3.8 ± 2.8	1.2 ± 1.1	0.4 ± 0.9	4.0 ± 1.2	6.0 ± 1.2	2.6 ± 1.5	6.8 ± 2.3	5.2 ± 1.8	$\mathbf{10.6} \pm 3.3$

Overall, DSP performs competitively on stitching tasks and substantially improves over flat policy baselines on most environments. With task-specific guidance, DSP achieves the best or near-best performance in several settings, including `pointmaze-large`, `pointmaze-teleport`, `antmaze-giant`, `humanoidmaze-giant`, and both `antsoccer` stitching tasks. These results suggest that guided generative subgoal planning can support trajectory composition when the learned subgoal distribution captures sufficient local connectivity.

The comparison between DSP$_1$ and DSP shows that guidance strength is important in stitching regimes. In many larger or more dynamically complex environments, using the task-specific guidance scale improves performance over standard conditional sampling, suggesting that properly tuned goal-directed guidance helps select more useful subgoals from the learned conditional distribution. This is consistent with prior diffusion-based decision-making work suggesting that conditional generative models can exhibit implicit dynamic-programming-like behavior through trajectory or subtrajectory generation [2, 31]. In DSP, this effect appears at the subgoal level: CFG amplifies the goal-conditioned component of the learned subgoal distribution, helping compose locally supported transitions into longer-horizon behavior.

At the same time, DSP does not uniformly dominate Bellman-backup-based hierarchical methods. HIQL and HIQL$^{\mathrm{w/o}}$ remain stronger on some `antmaze` and `humanoidmaze` stitch tasks, likely because value-based methods can propagate goal information across trajectory fragments through Bellman backups. By contrast, DSP relies more directly on the transition structure captured by its generative subgoal model. Thus, the stitching results show that DSP can perform effective trajectory composition in several regimes, rather than replacing Bellman-backup-based stitching in all settings.

Finally, the DSP$^{\mathrm{mix}}$ variants indicate that the default OGBench actor goal mixing strategy is not always suitable for generative high-level planning. While random goal relabeling can benefit value-based policy extraction, excessive random goals may shift the learned conditional distribution toward broad connectivity rather than directional, goal-relevant subgoals. The structured goal sampling used by DSP better preserves the goal-directed signal amplified by CFG, highlighting the interaction between goal sampling and inference-time guidance.

M Data Coverage and Behavior Quality

M.1 Reduced-Data Coverage

Using a fixed subset seed, we construct nested datasets by randomly ordering complete trajectories and retaining them until reaching each target data fraction. This preserves trajectory boundaries and future-state relabeling, and all methods use identical subsets.

OGBench evaluates AntMaze and HumanoidMaze success using XY position with a tolerance of 0.5. We therefore define 0.5-unit bins as $b_{0.5}(x, y) = (\lfloor x/0.5 \rfloor, \lfloor y/0.5 \rfloor)$ and measure

$$C_{\mathrm{XY}}(\mathcal{D}_r) = \frac{|\{b_{0.5}(x_t, y_t) : (x_t, y_t) \in \mathcal{D}_r\}|}{|\{b_{0.5}(x_t, y_t) : (x_t, y_t) \in \mathcal{D}_{100}\}|}. \tag{71}$$

C_{XY} measures relative spatial occupancy, not full-state coverage. Data fraction additionally affects visitation density, local transitions, and same-trajectory future-state pairs.

Table 9: Success rates (%) under nested complete-trajectory subsets. Results are mean $\pm$ standard deviation over five seeds. C_{XY} denotes relative XY-bin coverage.

Datasets	Fraction	C_{XY}	HIQL	HIQL$^{\mathrm{w/o}}$	DSP
`AntMaze-Large`	1.00	1.00	87.8 ± 1.5	88.6 ± 2.1	$\mathbf{92.0} \pm 4.1$
	0.50	0.96	87.0 ± 4.3	81.4 ± 3.2	$\mathbf{90.2} \pm 1.9$
	0.25	0.91	79.8 ± 2.7	70.2 ± 5.1	$\mathbf{83.0} \pm 2.1$
	0.10	0.85	19.4 ± 4.7	39.6 ± 5.6	$\mathbf{52.8} \pm 6.6$
`HumanoidMaze-Giant`	1.00	1.00	5.0 ± 1.6	0.0 ± 0.0	$\mathbf{66.0} \pm 3.4$
	0.10	0.94	1.0 ± 0.7	0.0 ± 0.0	$\mathbf{44.4} \pm 4.2$
	0.01	0.50	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{5.6} \pm 3.0$

DSP achieves the highest mean at every evaluated fraction. However, it drops from 92.0 to 52.8 on AntMaze-Large and from 66.0 to 5.6 on HumanoidMaze-Giant, confirming that severe loss of data support remains a failure condition. These results support DSP's relative competitiveness under the evaluated reduced-data settings, rather than coverage independence.

M.2 Official Explore Datasets

The official OGBench Explore datasets contain non-goal-directed trajectories: the locomotion direction is randomly resampled every 10 steps, while actions are produced by a pretrained Ant controller and perturbed with action noise of magnitude 1.0 [57]. They therefore test poor global trajectory quality while retaining locally executable transitions.

DSP achieves the highest mean on Medium and Large, but not on Teleport. Teleport randomly sends the agent to one of multiple exits, including a dead end; hence, a future state observed in the data is not necessarily an outcome the policy can reliably select. Future-state relabeling may therefore include favorable but uncontrollable outcomes, while replanning can only respond after the realized

Table 10: Success rates (%) on the official AntMaze Explore datasets, reported as mean ± standard deviation over five seeds.

Datasets	HIQL	HIQL$^{\text{w/o}}$	DSP
antmaze-medium-explore-v0	32.4 ± 10.0	33.4 ± 9.7	$\mathbf{60.4} \pm 9.3$
antmaze-large-explore-v0	1.6 ± 3.6	6.0 ± 6.7	$\mathbf{16.4} \pm 7.5$
antmaze-teleport-explore-v0	26.8 ± 16.6	$\mathbf{30.4} \pm 7.3$	21.4 ± 4.6

exit is observed. The result identifies stochastic, action-uncontrollable transitions as a limitation, rather than establishing uniform robustness across all Explore settings.

N Evaluation with Visual Observations

We evaluate DSP on four visual OGBench tasks using only $64 \times 64 \times 3$ RGB observations. Following the OGBench visual protocol, HIQL and DSP use the same IMPALA encoder architecture. DSP does not perform diffusion directly in pixel space; instead, it generates subgoals in the IMPALA latent space. HIQL additionally maps encoded state–subgoal pairs to its fixed 10-dimensional subgoal representation.

Table 11: Success rates (%) on visual OGBench tasks, reported as mean ± standard deviation over five seeds. Bold indicates the highest mean in each row.

Datasets	HIQL	DSP
visual-antmaze-medium-navigate	92.0 ± 2.1	$\mathbf{96.2} \pm 2.5$
visual-antmaze-large-navigate	51.8 ± 5.5	$\mathbf{68.4} \pm 6.2$
visual-antmaze-giant-navigate	4.4 ± 2.8	$\mathbf{13.8} \pm 4.4$
visual-scene-play	43.8 ± 3.9	$\mathbf{48.6} \pm 1.5$

As shown in Table 11, DSP achieves higher mean success on all four tasks, including improvements of 16.6 and 9.4 percentage points on Visual AntMaze-Large and AntMaze-Giant, respectively. These results show that DSP can be applied to visual observations through a shared latent encoder, without requiring diffusion directly in pixel space.

O Visualization of Subgoal Trajectories in Maze Environments

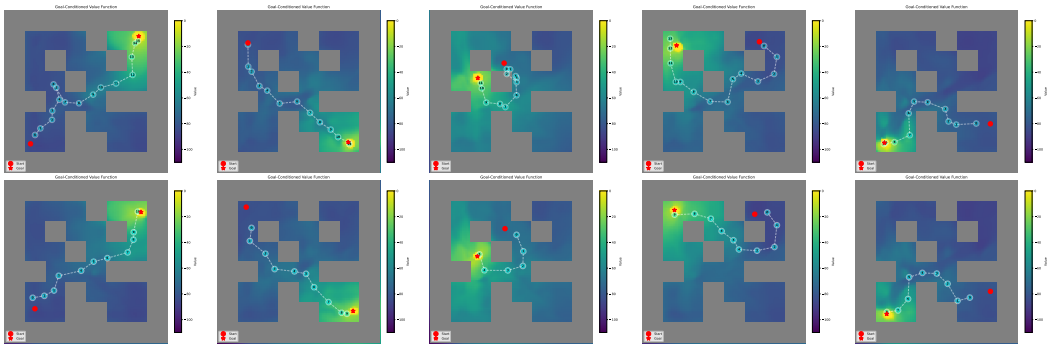
In this section, we present additional visualizations comparing the subgoal trajectories of HIQL$^{\text{w/o}}$ and DSP across the maze environments. Specifically, we display the planning behaviors for all five evaluation tasks, followed by a detailed analysis.

O.1 Medium Maze

We first visualize the results in the `medium`-sized maze. As shown in Figure 7, while both methods generally reach the goals, we observe distinct behavioral differences. HIQL$^{\text{w/o}}$ occasionally exhibits artifacts characteristic of value-based planning, such as suboptimal detours or subgoals projected into unreachable areas (e.g., walls). In contrast, DSP produces trajectories that are geometrically consistent and better follow the maze topology, generating smooth paths even in this shorter-horizon setting.

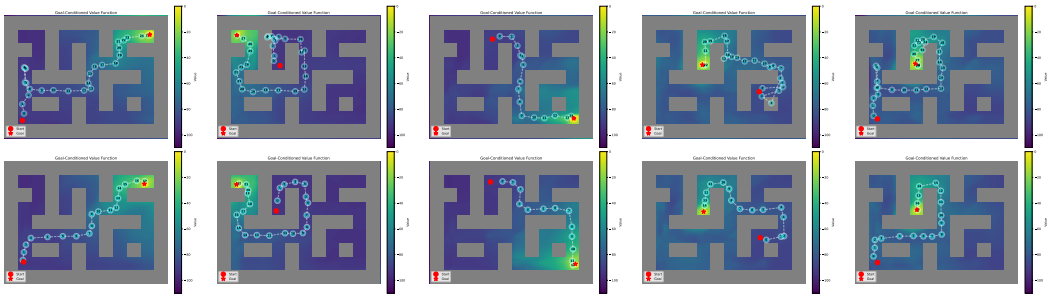
O.2 Large Maze

Next, we examine the `large`-sized maze, where the planning horizon increases. As illustrated in Figure 8, the impact of the horizon becomes visible. HIQL$^{\text{w/o}}$ begins to manifest noticeable detours and inefficient paths, showing less consistent planning as the scale increases. Although the value function eventually corrects the agent's direction, the guidance is noisy. DSP, however, maintains its planning stability, producing topologically coherent and smooth trajectories without the jaggedness observed in the baseline.



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4 (e) Task 5

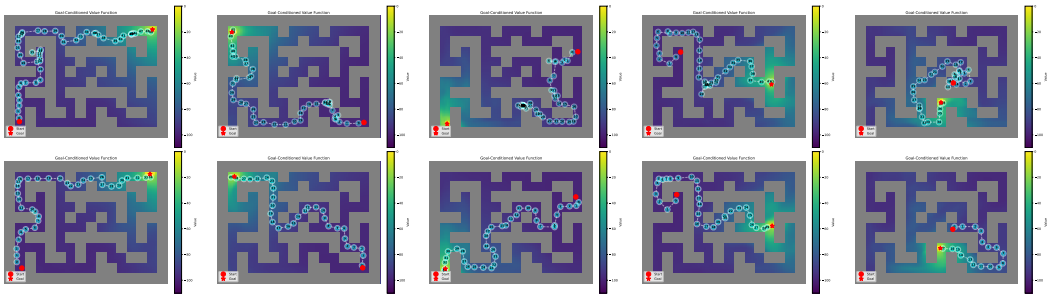
Figure 7: Visualization in the `medium` maze. Top: HIQL$^{\text{w/o}}$, Bottom: DSP. DSP generates smoother trajectories that better follow the topology.



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4 (e) Task 5

Figure 8: Visualization in the `large` maze. Top: HIQL$^{\text{w/o}}$, Bottom: DSP. As the scale increases, HIQL$^{\text{w/o}}$ begins to show significant detours, whereas DSP maintains efficient planning.

O.3 Giant Maze



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4 (e) Task 5

Figure 9: Visualization in the `giant` maze. Top: HIQL$^{\text{w/o}}$, Bottom: DSP. In this extreme setting, HIQL$^{\text{w/o}}$ often produces less reliable plans, while DSP more consistently reaches the target.

Finally, we extend our visualization to the `giant`-sized maze, where the curse of horizon is most acute. Here, HIQL$^{\text{w/o}}$ exhibits severe performance degradation. The high-level policy, struggling with the immense scale, often fails to generate valid subgoals, leading to task failure. Even in successful trials, the attenuation of the value signal results in subgoals that linger in local optima; although the agent may eventually escape, the traversal is far from time-optimal.

In sharp contrast, DSP shows more consistent navigation behavior. However, in this extreme setting, we observe a phenomenon of **subgoal clustering**, where consecutive high-level subgoals are repeatedly generated in a local region. Empirically, this behavior is not indicative of high-level planning failure, as the generated subgoals remain topologically valid and goal-directed. Instead, the clustering pattern demonstrates the planner's robustness to local execution stochasticity. Since the low-level policy is trained via AWR, it may exhibit minor deviations due to local value approximations. Crucially, unlike the larger global planning errors seen in baselines, these deviations are bounded within short horizons. As a result, the high-level planner successfully re-issues corrective subgoals in the same vicinity to compensate for these local execution errors, ensuring eventual task completion.

P Comparison Experiments on D4RL Antmaze

To further position DSP with respect to conceptually related diffusion-based hierarchical planning methods, we additionally evaluate DSP on the D4RL AntMaze benchmark. Our main experiments are conducted on OGBench, which is specifically designed for offline goal-conditioned RL and provides broader coverage of long-horizon reasoning, stitching, and stochasticity. By contrast, D4RL AntMaze is more limited for evaluating offline GCRL since it mainly uses a single fixed goal. Nevertheless, D4RL AntMaze is the benchmark on which several prior diffusion-based hierarchical planning methods were originally reported, and therefore provides the most direct setting for comparison.

Table 12: Comparison with diffusion-based hierarchical planning methods on D4RL AntMaze. We report mean $\pm$ standard deviation over 5 random seeds, with each seed evaluated on 100 episodes. Baseline numbers are taken from the corresponding original papers when reported under the same D4RL AntMaze setting. The best-performing entry in each row is highlighted in **bold**.

Datasets	HDMI	DTAMP	HD	SIHD	DSP
antmaze-umaze-v2	86.1 ± 2.4	–	–	–	$\mathbf{98.6 \pm 2.1}$
antmaze-umaze-diverse-v2	73.7 ± 1.1	–	94.0 ± 4.9	96.5 ± 2.8	$\mathbf{96.8 \pm 2.6}$
antmaze-medium-play-v2	–	$\mathbf{89.3 \pm 3.9}$	–	–	87.0 ± 3.3
antmaze-medium-diverse-v2	–	76.7 ± 4.5	88.7 ± 8.1	92.2 ± 5.0	$\mathbf{92.4 \pm 4.4}$
antmaze-large-play-v2	–	62.0 ± 2.9	–	–	$\mathbf{88.8 \pm 4.1}$
antmaze-large-diverse-v2	71.5 ± 3.5	53.3 ± 9.7	83.6 ± 5.8	89.4 ± 4.2	$\mathbf{95.4 \pm 2.9}$

Table 12 compares DSP with representative diffusion-based hierarchical planning baselines, including HDMI [38], DTAMP [24], HD [6], and SIHD [74]. We report mean and standard deviation over 5 random seeds, with each seed evaluated on 100 episodes. Baseline numbers are taken from the corresponding original papers when reported on the same D4RL AntMaze setting. The best-performing entry in each row is highlighted in bold.

Overall, DSP is highly competitive and outperforms prior diffusion-based hierarchical planning methods on most reported tasks. The gains are particularly pronounced on larger and more challenging tasks such as `antmaze-large-play-v2` and `antmaze-large-diverse-v2`, where long-horizon decision errors are more likely to accumulate. On averages over overlapping tasks, DSP substantially outperforms DTAMP (90.9 vs. 70.3) and HDMI (96.9 vs. 77.1), and also improves over HD (94.9 vs. 88.8) and SIHD (94.9 vs. 92.7).

These results are consistent with the key design of DSP. Unlike prior diffusion-based hierarchical planning methods, DSP does not use diffusion as a trajectory- or sequence-level planner. Instead, it models high-level decision making as a goal-conditioned generative subgoal policy. This is consistent with the reduced high-level value-noise path analyzed in Proposition 4.2. At the same time, classifier-free guidance enables direct and controllable goal-directed subgoal selection at inference time, while Proposition 5.1 explains the implicit advantage-weighted bias induced by this guidance.

Q Dexterous Manipulation Evaluation

We evaluate DSP and HIQL on two bimanual dexterous-manipulation tasks from Bi-DexHands [8]: `ShadowHandOver` and `ShadowHandCatchUnderarm`. Using the official collection interface, we construct a fixed offline dataset of 10^6 transitions per task with PPO. Both methods are trained on identical datasets under the same OGBench-style GCRL protocol. HIQL and DSP results are

success rates (%) reported as means $\pm$ standard deviations over five seeds. PPO is shown only as a data-collection reference, not as an offline or compute-matched baseline.

The tuples in Table 13 report (d_s, d_a, d_w, d_g). The 7D goal g is derived directly from the task-specified target-object pose, while the original 7D waypoint w is a future achieved object pose. To isolate the effect of waypoint dimensionality without changing the final task, we keep the 7D goal fixed and replace the task-space waypoint with a 55/67D task-relevant configuration-space waypoint containing bimanual configuration and object-pose information. Here, d_w denotes the waypoint dimension before method-specific processing: HIQL retains its learned subgoal representation, whereas DSP directly generates in the selected waypoint space.

Table 13: Success rates (%) on Bi-DexHands. PPO denotes the data collector. Dimensions are reported as (d_s, d_a, d_w, d_g).

Datasets	(d_s, d_a, d_w, d_g)	PPO	HIQL	DSP
`ShadowHandOver`	$(387, 40, 7, 7)$	46.5	56.6 ± 4.2	$\mathbf{74.2} \pm 2.9$
`ShadowHandCatchUnderarm`	$(411, 52, 7, 7)$	37.2	46.0 ± 4.1	$\mathbf{54.6} \pm 3.3$
`ShadowHandOver`	$(387, 40, 55, 7)$	46.5	54.2 ± 5.9	$\mathbf{67.6} \pm 2.4$
`ShadowHandCatchUnderarm`	$(411, 52, 67, 7)$	37.2	41.2 ± 3.4	$\mathbf{46.4} \pm 3.9$

With 7D task-space waypoints, DSP achieves higher mean success than HIQL by 17.6 and 8.6 percentage points on `ShadowHandOver` and `ShadowHandCatchUnderarm`, respectively. Moving to 55/67D configuration-space waypoints reduces DSP's mean success by 6.6/8.2 points, compared with 2.4/4.8 points for HIQL. Nevertheless, DSP retains higher mean success by 13.4/5.2 points in the two higher-dimensional waypoint settings. These results show that waypoint dimensionality is a genuine challenge for direct DSP generation, while providing initial task-specific evidence that its advantage is not solely attributable to the 7D waypoint representation.

The collected datasets contain executable local behaviors. Accordingly, these experiments compare how DSP and HIQL plan and execute bimanual behavior under identical offline data support, rather than testing whether either method can acquire skills absent from the dataset. The results provide initial evidence beyond OGBench's parallel-jaw manipulation setting, without constituting a comprehensive evaluation of dexterous manipulation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately state our core contribution of proposing Diffusion Subgoal Planning (DSP) to replace value-based high-level guidance with guided generative inference, which is consistently supported by the theoretical analysis in Sections 4 and 5, and the empirical results in Section 6.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We explicitly discuss our limitations in Appendix A, including sensitivity to limited data coverage and out-of-distribution goals, and the inference cost introduced by iterative diffusion sampling.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All assumptions for our theoretical results (e.g., the 1D toy environment and score function approximation) are explicitly stated within their respective proposition statements, and their complete formal proofs are provided in Appendix B with clear cross-references in Sections 4 and 5.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide complete algorithm pseudocode in Section 5.4, detailed network architectures in Appendix C, and exhaustive hyperparameter configurations (both common and task-specific) in Appendix D. Additionally, the baselines in our main OGBench experiments are rerun using their released code. The D4RL baseline results in Table 12 are taken from the original papers.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide a code repository, which includes scripts and instructions to reproduce our main OGBench results and the D4RL AntMaze comparisons. The OGBench and D4RL datasets are publicly available (OGBench [57] and D4RL [14]).

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We detail the optimizer, network architectures, and goal sampling strategies in Appendix C, with exhaustive common and task-specific hyperparameters provided in Appendix D. Evaluation protocols, including the number of seeds and test-time goals, are specified in Section 6.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: We report mean and 1-sigma standard deviations across 5 random seeds for all main experimental results (Tables 1, 5, 8, and 12) and hyperparameter sensitivity analyses (Figure 5). The reported variability corresponds to the standard deviation over 5 random seeds after averaging performance across the evaluation goals or episodes for each task.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We specify that all experiments were conducted on two compute servers, each equipped with four NVIDIA GeForce RTX 4090 GPUs (24 GB of VRAM per GPU) and report exact training times for all methods in Table 7, with the largest locomotion task requiring approximately one hour.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work is foundational research in offline goal-conditioned reinforcement learning using publicly available simulated benchmarks. It involves no human subjects, sensitive data, or ethically sensitive applications.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This paper presents foundational algorithmic research in offline goal-conditioned reinforcement learning, evaluated exclusively on standard simulated benchmarks. Because the work is not tied to a particular real-world application or deployment, it has no direct immediate societal impact, and a broader impacts discussion was therefore omitted from the manuscript.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work proposes an offline goal-conditioned reinforcement learning algorithm evaluated on standard, publicly available simulated benchmarks. We do not release pre-trained models with high misuse potential (such as language or image generation models) nor scraped datasets, so no specific safeguards are required.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We utilize publicly available benchmarks (OGBench, D4RL) and baseline algorithms, all of which are properly cited in the paper. In our code repository, we retain the original license files and READMEs for these assets to ensure proper attribution and compliance with their terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the implementation of our proposed algorithm, Diffusion Subgoal Planning (DSP), as a new asset in our code repository. This repository includes structured documentation, environment setup instructions, and scripts required to reproduce our main results.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve any crowdsourcing or research with human subjects. All experiments are conducted using automated agents in simulated environments (OGBench and D4RL).

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve any research with human subjects, so IRB approval or equivalent review was not required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core methodology of this paper relies on diffusion models and offline reinforcement learning algorithms, and does not involve Large Language Models (LLMs) as any component of the method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.