TRAJECTORY EDITING AS POSTERIOR INFERENCE FOR OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline RL performance depends on the quality of trajectories in a fixed dataset. Nevertheless, practical datasets often contain structural deficiencies along two axes: spatial gaps, where useful locomotion segments lack connecting transitions, and temporal sparsity, where manipulation demonstrations under-sample critical phases such as contact, grasping, and alignment. These deficiencies limit downstream policy learning, reducing returns or success rates. We introduce trajectory editing to repair offline data without environment interaction by bridging disconnected segments or densifying critical phases. The key challenge is to satisfy the prescribed anchors of each edit while remaining within the support of the offline dataset and consistent with its dynamics. We propose **B**ayesian **I**nference for **T**rajectory **E**diting **(BITE)**, a unified framework that treats the edited trajectory as a latent variable to be inferred. BITE learns a prior over plausible trajectories from the offline dataset and constructs an anchor-consistency likelihood from the edit specification, which induces a posterior that trades off data plausibility and edit satisfaction. We instantiate this prior with a diffusion trajectory model and derive a lifted posterior sampler that propagates clean-level edit observations through the reverse diffusion chain using backward sequential Monte Carlo. The resulting posterior inference procedure achieves sampling consistency as the particle population grows and its edit posteriors are provably as accurate as the learned prior, with no dependence on any edit distribution. Experiments on `D4RL` locomotion and `RoboMimic` manipulation tasks show that BITE delivers the strongest performance gains among baselines. Our code is publicly available at `https://anonymous.4open.science/r/bite-055D`.

1 INTRODUCTION

Offline reinforcement learning (RL) has become an important paradigm for sequential decision-making, where policies are learned solely from previously collected trajectories without online interaction with the environment (Levine et al., 2020). This setting is attractive in real-world applications where trial-and-error interaction is costly, unsafe, or unavailable, and provides a valuable path for scaling RL in domains such as autonomous driving (Lee et al., 2024a), robotics (Gürtler et al., 2023), and healthcare (Melehy et al., 2026). Nevertheless, this benefit comes with a constraint: because no additional data are collected during training, the coverage, consistency, and behavioral diversity of the fixed trajectories directly determine what policies can be learned.

In practice, logged datasets are not produced by uniform exploration. They inherit biases from behavior policies, task objectives, and logging pipelines (Fu et al., 2020; Schweighofer et al., 2022; Hong et al., 2023b; Zheng et al., 2023); as a result, they are concentrated in particular regions, skills, and time scales. This concentration can create structural deficiencies along two axes. Spatially, locomotion data may contain useful trajectory segments separated by missing transition evidence, preventing offline learners from composing them into coherent high-return behavior (Li et al., 2024). Temporally, manipulation demonstrations may under-sample short but decisive execution phases, such as contact, grasping, or insertion, leaving too few intermediate transitions for learning precise control. Figure 1 visualizes the trajectory-level manifestations: spatial gaps can appear as drops in locomotion stability, while temporal sparsity can appear as localized action chaos around precision-sensitive manipulation phases.

These observations motivate us to edit offline trajectories to repair structural deficiencies without additional environment interaction. In the spatial case this means bridging disconnected behavior fragments; in the temporal case it means retiming demonstrations by expanding key phases of

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

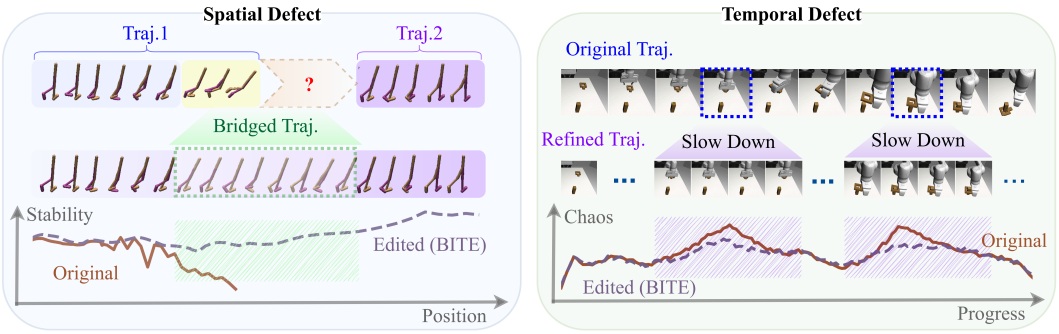


Figure 1: **Left:** Spatial gaps in locomotion data leave useful trajectory segments disconnected due to missing transition evidence. BITE bridges these gaps, yielding edited trajectories with improved stability. **Right:** Temporal sparsity in manipulation demonstrations under-samples precision-sensitive phases, leading to localized action chaos. BITE densifies these phases, reducing action chaos around critical execution stages. More details about the example can be found in Appendix J.1.

execution. However, satisfying the desired edit alone is not sufficient. Since the true environment dynamics are unknown and no additional interaction is allowed, edited trajectories must also remain compatible with the behavioral support and transition dynamics reflected in the offline data. Thus, trajectory editing must be both corrective and conservative: corrective enough to repair structural deficiencies, yet conservative enough to avoid generating out-of-support behavior.

To address this challenge, we propose **B**ayesian **I**nference for **T**rajectory **E**diting **(BITE)**, a unified framework for editing offline trajectories under structural constraints while preserving distributional plausibility. Our contributions are summarized as follows. **(1)** We identify two complementary sources of information from a probabilistic viewpoint: the offline dataset provides a data-driven prior over plausible trajectories, while the editing objective induces a likelihood that enforces structural compliance. Under this formulation, the edited trajectory is treated as a latent variable to be inferred, while data plausibility and edit consistency are naturally balanced through Bayes' rule. **(2)** We develop a diffusion-based posterior inference algorithm for edited trajectory generation. BITE learns a diffusion prior over state-action trajectories and augments it with a transition-consistency potential, yielding a tilted prior that favors locally feasible dynamics. Because edit constraints are defined at the clean-trajectory level, we introduce posterior lifting to propagate these constraints throughout the reverse diffusion process, allowing the edit signal to guide each denoising step. We further derive a backward sequential Monte Carlo sampler that recursively approximates the lifted posterior and prove its consistency in probability for the clean-level posterior marginal as the number of particles increases. **(3)** We show that Bayesian editing transfers prior approximation error directly to every edit posterior, whereas the error of amortized editors (Li et al., 2024) can grow without bound on rare edits and is uncontrolled on unseen ones. Empirically, BITE achieves state-of-the-art average normalized returns on `D4RL` locomotion tasks and consistently improves success rates on `RoboMimic` manipulation tasks across multiple offline RL algorithms.

2 Problem Formulation

We consider a Markov decision process (MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$, where $\mathcal{S} \subseteq \mathbb{R}^{d_s}$ and $\mathcal{A} \subseteq \mathbb{R}^{d_a}$ are the state and action spaces, $\mathcal{P}(\cdot|s,a)$ is the transition kernel, $r(s,a)$ is the reward function, and $\gamma \in [0,1)$ is the discount factor. A trajectory with length T is defined by $\tau = ((s_0,a_0,r_0),\dots,(s_{T-1},a_{T-1},r_{T-1}))$. In offline RL, the learner receives a fixed dataset $\mathcal{D} = \{\tau^{(i)}\}_{i=1}^{N_d}$ and trains a policy without further environment interaction. We study trajectory editing as a way to repair structural deficiencies in $\mathcal{D}$. An editing instance is specified by (Ξ, ϵ), where Ξ is the source content and ϵ is the edit condition. The goal is to construct an edited trajectory $\xi = \big((\tilde{s}_0,\tilde{a}_0,\tilde{r}_0),\dots,(\tilde{s}_{H-1},\tilde{a}_{H-1},\tilde{r}_{H-1})\big)$ with horizon H. Since edit constraints are imposed on states and actions, we write $\varphi((s_t,a_t,r_t)) = (s_t,a_t)$ and extend φ elementwise to trajectories.

Spatial Editing. Spatial editing connects two trajectory fragments by inferring a bridge between them. The source content is an ordered pair of anchor fragments $\Xi_{\text{se}} = (\tau^a_{0:T_a-1}, \tau^b_{0:T_b-1})$, and the edit condition $\epsilon_{\text{se}} = \Delta$ specifies the bridge length. The edited horizon is $H = T_a + \Delta + T_b$. A valid edited trajectory preserves the two anchors:

$$\varphi(\xi_{0:T_a-1}) = \varphi(\tau^a_{0:T_a-1}), \quad \varphi(\xi_{H-T_b:H-1}) = \varphi(\tau^b_{0:T_b-1}).$$



Figure 2: Bayesian perspective underlying trajectory editing.

The middle segment $\xi_{T_a:H-T_b-1}$ is inferred as the bridge. Rewards are not constrained, since inserting a bridge changes the transition context.

Temporal Editing. Temporal editing increases the temporal resolution of a trajectory by inserting intermediate transitions around key execution phases. The source content is a trajectory $\Xi_{\text{te}} = \tau_{0:T-1}$. The edit condition $\epsilon_{\text{te}} = (H, \kappa)$ specifies a target horizon $H > T$ and a monotone correspondence $\kappa : \{0, \dots, T-1\} \to \{0, \dots, H-1\}$. A valid edited trajectory preserves source states at the aligned anchors:

$$\tilde{s}_{\kappa(t)} = s_t, \quad t = 0, \dots, T-1.$$

Intermediate states are inferred to refine the original execution. Actions and rewards are not fixed, since a finer-grained execution may require different low-level controls and transition contexts.

Both spatial and temporal editing lead to the same central question: given an editing instance (Ξ, ϵ), how can we infer an edited trajectory ξ that satisfies the prescribed structural constraints while remaining compatible with the behavioral regularities reflected in the offline dataset $\mathcal{D}$?

3 Method

To address this question, we cast trajectory editing as Bayesian inference. The offline dataset induces a data-driven prior over plausible trajectories, while the source content Ξ and edit condition ϵ define a likelihood that enforces anchor consistency. The edited trajectory is then inferred as a latent variable under the resulting posterior; see Figure 2.

3.1 Prior Elicitation

We first learn a data-driven prior from the offline dataset $\mathcal{D}$. Let $x^{[0]} := \text{vec}(\varphi(\xi)) \in \mathbb{R}^{d_x}$, where $d_x = H(d_s + d_a)$. We learn the state-action prior with a diffusion model over $x^{[0]}$. Introducing noisy latent variables $x^{[1]}, \dots, x^{[L]}$ along the diffusion chain, the reverse-time generative process is

$$p(x^{[0:L]}|\mathcal{D}) = p(x^{[L]}) \prod_{l=1}^{L} p(x^{[l-1]}|x^{[l]}, \mathcal{D}), \tag{1}$$

where $x^{[l]}$ denotes the noisy trajectory at diffusion level l. The marginal distribution of $x^{[0]}$ defines the learned state-action prior $p(x^{[0]}|\mathcal{D})$.

After editing the state-action sequence, rewards are completed using a conditional reward model:

$$p(\tilde{r}_{0:H-1}|\varphi(\xi), \mathcal{D}) = \prod_{t=0}^{H-1} p(\tilde{r}_t|\tilde{s}_t, \tilde{a}_t, \mathcal{D}). \tag{2}$$

This factorization lets the diffusion prior focus on the state-action structure needed for editing, while rewards are predicted afterward as local labels, avoiding abrupt reward changes that are common in sparse-reward tasks (Andrychowicz et al., 2017).

The diffusion prior captures global trajectory structure, but it does not explicitly enforce one-step transition consistency. To incorporate local feasibility into the prior, we learn a transition model $f_{\mathcal{D}} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ from the offline dataset and define the transition-consistency potential

$$\phi_{\mathcal{D}}(x^{[0]}) = \exp\left(-\lambda \sum_{t=0}^{H-2} \|\tilde{s}_{t+1} - f_{\mathcal{D}}(\tilde{s}_t, \tilde{a}_t)\|_2^2\right), \tag{3}$$

where $\lambda > 0$ controls the strength of transition consistency.

We then define the tilted state-action prior $\bar{p}(x^{[0]}|\mathcal{D}) \propto p(x^{[0]}|\mathcal{D})\phi_{\mathcal{D}}(x^{[0]})$ and the full trajectory prior used by BITE is therefore $\bar{p}(\xi|\mathcal{D}) = \bar{p}(x^{[0]}|\mathcal{D}) \prod_{t=0}^{H-1} p(\tilde{r}_t|\tilde{s}_t, \tilde{a}_t, \mathcal{D})$. Thus, the prior combines global plausibility from the diffusion model, local feasibility from the transition-consistency tilt, and reward completion from the conditional reward model.

3.2 LIKELIHOOD SPECIFICATION

In this subsection, we encode edit consistency as a likelihood function. Let $x^{[0]} := \mathrm{vec}(\varphi(\xi))$ be the clean state-action trajectory. For each editing instance (Ξ, ϵ), we construct an anchor observation $y^{[0]} = \mathcal{Y}(\Xi)$ and a linear editing operator $\mathcal{E}$ such that exact consistency is $y^{[0]} = \mathcal{E}x^{[0]}$. We relax this constraint with a Gaussian likelihood:

$$p(\Xi|\xi, \epsilon) = p(y^{[0]}|x^{[0]}) = \mathcal{N}\left(y^{[0]}; \mathcal{E}x^{[0]}, \sigma^2 I\right), \tag{4}$$

where σ^2 controls the allowed anchor mismatch.

Spatial editing. For spatial editing, the source content consists of two anchor fragments $\Xi_{\mathrm{se}} = (\tau^a_{0:T_a-1}, \tau^b_{0:T_b-1})$. The observation is $\mathcal{Y}(\Xi_{\mathrm{se}}) = \begin{bmatrix}\mathrm{vec}(\varphi(\tau^a_{0:T_a-1}))\\ \mathrm{vec}(\varphi(\tau^b_{0:T_b-1}))\end{bmatrix}$, and the editing operator selects the prefix and suffix of the edited trajectory: $\mathcal{E}_{\mathrm{se}} = \begin{bmatrix} I_{T_a} & 0_{T_a\times\Delta} & 0_{T_a\times T_b} \\ 0_{T_b\times T_a} & 0_{T_b\times\Delta} & I_{T_b}\end{bmatrix} \otimes I_{d_s+d_a}$. Thus, the two fragments are fixed as anchors, while the intermediate Δ steps are inferred.

Temporal editing. For temporal editing, the source content is $\Xi_{\mathrm{te}} = \tau_{0:T-1}$, and only source states are anchored. We set $\mathcal{Y}(\Xi_{\mathrm{te}}) = \mathrm{vec}(s_0, \ldots, s_{T-1})$. Given the correspondence κ, define $\mathcal{B}_\kappa \in \{0,1\}^{T\times H}$ by $[\mathcal{B}_\kappa]_{t,h} = \mathbf{1}\{h = \kappa(t)\}$. The editing operator is $\mathcal{E}_{\mathrm{te}} = \mathcal{B}_\kappa \otimes [I_{d_s} \;\; 0_{d_s\times d_a}]$. Thus, the prescribed source states are fixed as anchors at the aligned positions $\{\kappa(t)\}_{t=0}^{T-1}$ in the edited trajectory, while the remaining $H - T$ states and all H actions are inferred from the trajectory prior.

3.3 LIFTED POSTERIOR OVER THE DIFFUSION CHAIN

We next construct the posterior distribution over the diffusion chain. The likelihood in (4) and the transition-consistency potential in (3) constrain only the final denoised trajectory $x^{[0]}$ from the diffusion model. However, the state-action prior is represented by the reverse diffusion chain $x^{[L]} \to x^{[L-1]} \to \cdots \to x^{[0]}$ in (1). If both are applied only at level 0, the edit and transition-consistency signals enter the generative process too late to control earlier denoising steps, so we lift $p(y^{[0]}|x^{[0]})$ and $\phi_{\mathcal{D}}(x^{[0]})$ to the full diffusion path.

Lifted likelihood. For each diffusion level l, we construct an auxiliary observation $y^{[l]}$ by applying the same forward noising schedule to the clean observation (Dou & Song, 2024):

$$y^{[l]} = \sqrt{\bar\alpha_l}\, y^{[0]} + \sqrt{1-\bar\alpha_l}\, \mathcal{E}z, \quad z \sim \mathcal{N}(0, I_{d_x}), \tag{5}$$

where $\bar\alpha_l$ is the cumulative noise coefficient. This construction is consistent with the forward diffusion process in (Ho et al., 2020). If $x^{[l]} = \sqrt{\bar\alpha_l}x^{[0]} + \sqrt{1-\bar\alpha_l}z$ and $y^{[0]} = \mathcal{E}x^{[0]} + \eta$ with $\eta \sim \mathcal{N}(0, \sigma^2 I)$, then $y^{[l]} = \mathcal{E}x^{[l]} + \sqrt{\bar\alpha_l}\eta$ under the coupled forward-noising construction. Thus, the level-wise edit likelihood is

$$p(y^{[l]}|x^{[l]}) = \mathcal{N}\left(y^{[l]}; \mathcal{E}x^{[l]}, \bar\alpha_l\sigma^2 I\right), \quad l = 0, \ldots, L. \tag{6}$$

At $l = 0$, $\bar\alpha_0 = 1$, which recovers the clean likelihood in (4).

Lifted potential. Since transition residuals are meaningful only when the sample is close to a clean state-action trajectory, we apply this guidance only near the clean end of the diffusion chain (i.e., $l \le L_{\mathrm{tc}}$):

$$\phi^{[l]}_{\mathcal{D}}(x^{[l]}) = \exp\left(-\lambda\mathbf{1}\{l \le L_{\mathrm{tc}}\}\textstyle\sum_{t=0}^{H-2}\left\|\tilde s^{[l]}_{t+1} - f_{\mathcal{D}}(\tilde s^{[l]}_t, \tilde a^{[l]}_t)\right\|_2^2\right). \tag{7}$$

Lifted posterior. Bayes' rule then gives the lifted state-action posterior

$$\pi_0(x^{[0:L]}) \propto \underbrace{p(x^{[L]})\textstyle\prod_{l=1}^{L} p(x^{[l-1]}|x^{[l]}, \mathcal{D})\prod_{l=0}^{L}\phi^{[l]}_{\mathcal{D}}(x^{[l]})}_{\text{lifted tilted prior}} \underbrace{\textstyle\prod_{l=0}^{L} p(y^{[l]}|x^{[l]})}_{\text{lifted likelihood}}. \tag{8}$$

Note that the marginal of $\pi_0(x^{[0:L]})$ on $x^{[0]}$ defines the clean-level posterior over edited state-action trajectories. After sampling $x^{[0]}$, rewards are completed using the conditional reward model in (2).

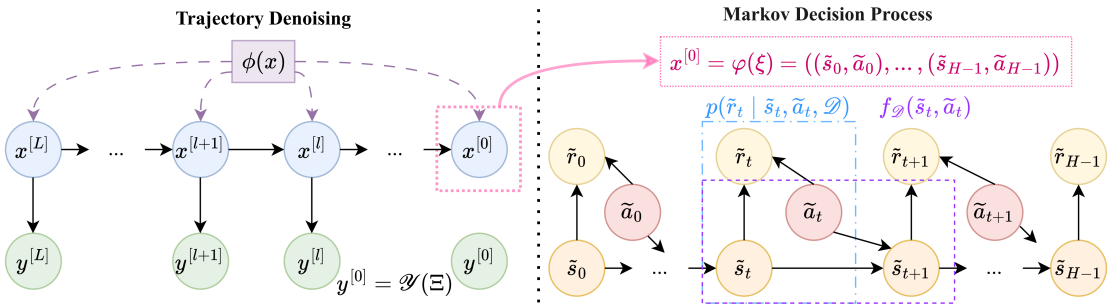


Figure 3: **Probabilistic graphical model of BITE.** The lifted tilted prior combines the reverse diffusion chain with transition-consistency potentials, while the lifted observations $\{y^{[l]}\}_{l=0}^{L}$ provide edit-consistency likelihoods at every denoising level. Rewards are completed after sampling the edited state-action trajectory.

3.4 Sequential Posterior Sampling

We now approximate the lifted posterior in (8). Because edit likelihoods and transition potentials are attached to every denoising level, we develop a backward sequential Monte Carlo sampler (Del Moral et al., 2006; Wu et al., 2023; Cardoso et al., 2024) that propagates particles from $x^{[L]}$ to $x^{[0]}$. Define the intermediate target at level l as

$$\pi_l(x^{[l:L]}) \propto p(x^{[L]}) \prod_{k=l+1}^{L} p(x^{[k-1]}|x^{[k]}, \mathcal{D}) \prod_{k=l}^{L} \left[\phi_{\mathcal{D}}^{[k]}(x^{[k]}) p(y^{[k]}|x^{[k]})\right].$$

The final target $\pi_0(x^{[0:L]})$ is exactly the lifted posterior in (8). These targets satisfy the backward recursion

$$\pi_{l-1}(x^{[l-1:L]}) \propto \pi_l(x^{[l:L]}) p(x^{[l-1]}|x^{[l]}, \mathcal{D}) p(y^{[l-1]}|x^{[l-1]}) \phi_{\mathcal{D}}^{[l-1]}(x^{[l-1]}). \tag{9}$$

At each level, the reverse transition proposes the next trajectory, the likelihood enforces the edit observation, and the potential regularizes transition consistency.

Suppose that after resampling at level l, we have particles $\{x^{[l](j)}\}_{j=1}^{M}$ approximating the level-l marginal. Given a parent particle $x^{[l](j)}$, we sample from the edit-guided proposal

$$x^{[l-1](j)} \sim q_{l-1}(\cdot|x^{[l](j)}, y^{[l-1]}) = p(\cdot|x^{[l](j)}, \mathcal{D}) p(y^{[l-1]}|\cdot) / m_{l-1}(x^{[l](j)}), \tag{10}$$

where $m_{l-1}(x^{[l]})$ is the predictive edit likelihood that normalizes the proposal. The proposal in (10) has a Gaussian form because the reverse diffusion transition is Gaussian (Ho et al., 2020) and the edit likelihood is linear-Gaussian (Cardoso et al., 2024; Dou & Song, 2024). The corresponding importance weight is

$$w_{l-1}^{(j)} \propto \phi_{\mathcal{D}}^{[l-1]}(x^{[l-1](j)}) \frac{p(x^{[l-1](j)}|x^{[l](j)}, \mathcal{D}) p(y^{[l-1]}|x^{[l-1](j)})}{q_{l-1}(x^{[l-1](j)}|x^{[l](j)}, y^{[l-1]})}.$$

Substituting the normalized proposal in (10) yields

$$w_{l-1}^{(j)} \propto \phi_{\mathcal{D}}^{[l-1]}(x^{[l-1](j)}) m_{l-1}(x^{[l](j)}). \tag{11}$$

The predictive likelihood $m_{l-1}(x^{[l](j)})$ favors parent particles that can produce anchor-consistent descendants, while $\phi_{\mathcal{D}}^{[l-1]}(x^{[l-1](j)})$ favors sampled children with locally feasible transitions.

After normalizing the weights, the particles are resampled before proceeding to the next denoising level. At level 0, one state-action trajectory is sampled from the final particle set, and rewards are generated from (2). We call the overall method **B**ayesian **I**nference for **T**rajectory **E**diting (BITE). The complete BITE algorithm is summarized in Algorithm 1 in Appendix A.

4 Theoretical Analysis

4.1 Posterior Sampling Consistency

The BITE editing stage uses a backward sequential sampler to approximate the lifted posterior in (8). We can prove that the final empirical distribution is consistent for the clean-level posterior marginal.

Theorem 1. *Let Π_0 be the probability measure associated with the lifted posterior π_0, and let $\{x^{[0](j)}\}_{j=1}^M$ be the level-0 particles returned by BITE after the final resampling step. Then*

$$\frac{1}{M}\sum_{j=1}^M \delta_{x^{[0](j)}} \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_0 \circ (x^{[0]})^{-1},$$

where $\Pi_0 \circ (x^{[0]})^{-1}$ denotes the marginal distribution of Π_0 on $x^{[0]}$, and $\xrightarrow{\mathbb{WP}}$ denotes weak convergence in probability.

The proof of this theorem can be found in Appendix B.

4.2 Bayesian Editing versus Amortized Editing

We next clarify the distinction between Bayesian editing and amortized editing, which also helps position BITE relative to related trajectory-stitching methods (Li et al., 2024). BITE belongs to the Bayesian editing paradigm: it learns a reusable unconditional trajectory prior $p_\theta(\xi)$ from the offline dataset, and each edit instance (Ξ, ϵ) is imposed through an explicit anchor-consistency likelihood $p(\Xi|\xi, \epsilon)$. In contrast, amortized editing methods train a conditional editor $p_\phi(\xi|\Xi, \epsilon)$ on edit instances from a training distribution ρ, with the editing constraint amortized into the learned generator. Let $p^\star$ be the data-generating trajectory distribution, $p^\star(\xi|\Xi, \epsilon) \propto p(\Xi|\xi, \epsilon)p^\star(\xi)$ the oracle posterior, and $m^\star(\Xi|\epsilon) = \int p(\Xi|\xi, \epsilon)\, p^\star(\xi)\, d\xi$ the distribution of edit instances induced by data-consistent trajectories; $p_\theta(\xi|\Xi, \epsilon)$ and m_θ are defined analogously from p_θ.

Assumption 1 (Bayesian prior)**.** $D_{\mathrm{KL}}(p^\star \,\|\, p_\theta) \le \delta$.

Assumption 2 (Amortized editor)**.** $\mathbb{E}_{(\Xi,\epsilon)\sim\rho}\big[D_{\mathrm{KL}}(p^\star(\cdot|\Xi, \epsilon) \,\|\, p_\phi(\cdot|\Xi, \epsilon))\big] \le \delta$.

Assumption 1 is what diffusion training minimizes: since $\log p_\theta \ge \mathrm{ELBO}_\theta$, averaging over $p^\star$ gives $D_{\mathrm{KL}}(p^\star\|p_\theta) \le \mathcal{L}(\theta) - H(p^\star)$, with $\mathcal{L}(\theta)$ the negative-ELBO training loss and H the differential entropy (Ho et al., 2020; Song et al., 2021). Assumption 2 is what conditional maximum-likelihood or ELBO training under ρ certifies.

Proposition 1. (i) *Under Assumption 1, for edit instances drawn as $\Xi \sim m^\star(\cdot|\epsilon)$ with ϵ following any fixed distribution over edit conditions,*

$$\mathbb{E}_{(\Xi,\epsilon)\sim m^\star}\big[D_{\mathrm{KL}}(p^\star(\cdot|\Xi, \epsilon) \,\|\, p_\theta(\cdot|\Xi, \epsilon))\big] = D_{\mathrm{KL}}(p^\star\|p_\theta) - D_{\mathrm{KL}}(m^\star\|m_\theta) \le \delta. \tag{12}$$

(ii) *Under Assumption 2,*

$$\mathbb{E}_{(\Xi,\epsilon)\sim m^\star}\big[D_{\mathrm{KL}}(p^\star(\cdot|\Xi, \epsilon) \,\|\, p_\phi(\cdot|\Xi, \epsilon))\big] \le \left\|\tfrac{dm^\star}{d\rho}\right\|_\infty \delta, \tag{13}$$

where $\|dm^\star/d\rho\|_\infty$ is the largest ratio between the data-induced and the training edit distributions, and is infinite if $m^\star$ puts mass where ρ does not.

The proof is given in Appendix C. In Bayesian editing, the edit enters only through the fixed likelihood, so the prior error transfers to the posterior directly and no edit distribution appears in the bound. In amortized editing, the guarantee is an average over the training distribution ρ, so the error on an edit region B that is rare under ρ can be amplified by $1/\rho(B)$.

5 Related Work

Offline RL data augmentation. Spatial editing in BITE can serve as offline RL data augmentation. Existing methods differ in the unit of synthetic data they generate. TATU (Zhang et al., 2023) is a rollout-based method that learns a dynamics model, generates imagined rollouts, and truncates uncertain segments to reduce model error. Although this improves local coverage, compounding model error limits its ability to synthesize long coherent trajectories. SER (Lu et al., 2023) is a transition-tuple simulation method that models individual transitions and adds sampled tuples to the replay buffer. This increases transition diversity, but tuple-level simulation does not preserve trajectory-level structure or long-horizon coherence. Trajectory-level augmentation provides a complementary advantage: instead of generating isolated transitions or model rollouts, it can preserve long-horizon structure and compose useful behavior fragments already present in the offline dataset. DStitch (Li et al., 2024) follows this direction by connecting trajectory fragments, making it the closest method to our spatial editing setting. However, DStitch amortizes the stitching constraint into its generator by learning from stitched examples constructed during training, which belongs to amortized editing in Section 4.2. Its error control is therefore averaged over the training stitching

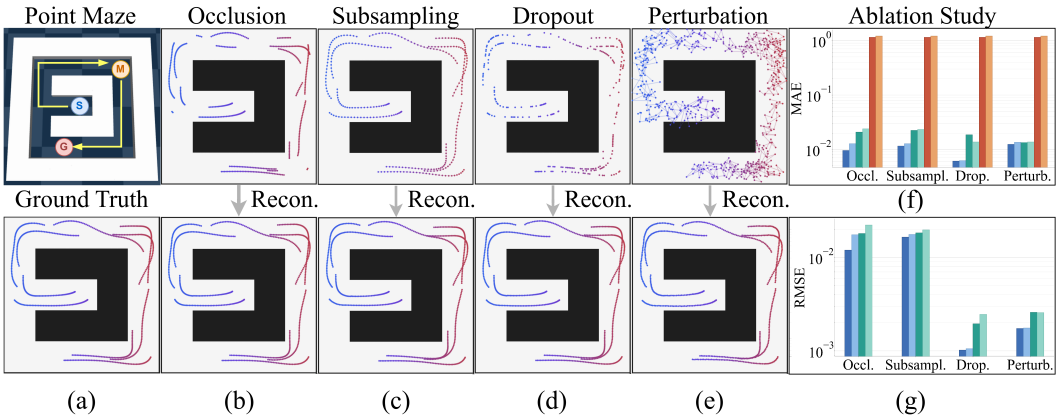


Figure 4: PointMaze reconstruction under four corruptions and corresponding ablations. Full w/ Dyn., Full w/o Dyn., Sparse w/ Dyn., Sparse w/o Dyn., Clean-only w/ Dyn., and Clean-only w/o Dyn. Missing bars indicate values that are too large to be meaningfully displayed.

distribution and can become weak on rare edit regions, whereas BITE imposes each concrete edit request at sampling time through an anchor-consistency likelihood on a reusable trajectory prior. Compared with DStitch, this Bayesian editing formulation transfers prior approximation accuracy to each queried edit posterior without amplification; see Proposition 1. The visualizations in Appendix G further validate our analysis: DStitch tends to produce abrupt zigzagging artifacts in the stitched segment, whereas BITE generates smoother bridges with better local continuity.

Image restoration with diffusion models. BITE is also related to diffusion-based image restoration, where a learned image prior is combined with corrupted-image evidence to recover a clean image. Representative methods differ in how the observed evidence is injected into restoration. DDRM (Kawar et al., 2022) uses the known corruption process to blend denoised predictions with measured image content. DPS (Chung et al., 2023) guides restoration by checking whether the current clean-image estimate, once corrupted again, matches the observation. FPS (Dou & Song, 2024) propagates corrupted-image evidence across multiple noise levels, allowing the observation to influence the restoration process throughout denoising. BITE has an analogous philosophy, but addresses a different trajectory editing problem to improve downstream offline RL performance.

6 Experiments

We organize the empirical evaluation around three questions. **Q1:** Can BITE repair the trajectories in a desired way? **Q2:** Can spatial editing improve the offline RL performance for locomotion tasks? **Q3:** Can temporal editing improve the offline RL performance for manipulation tasks?

6.1 Toy Example: Trajectory Reconstruction in PointMaze

We first study whether BITE can recover plausible trajectories from partial or corrupted observations. In PointMaze, we corrupt a G-shaped trajectory using four patterns: occlusion, subsampling, dropout, and perturbation. Occlusion (a missing contiguous segment) mimics spatial editing and subsampling (sparse observations) mimics temporal editing; dropout removes scattered observations and perturbation adds noise (details in Appendix D). As shown in Figure 4, BITE preserves the observed anchors and reconstructs smooth trajectories that follow the underlying maze topology across all corruption types.

Ablation Study. We ablate two inference components of BITE: full-chain likelihood guidance (6) and transition-consistency guidance (7). The former propagates the anchor-consistency signal through the entire reverse diffusion chain, while the latter encourages locally feasible one-step dynamics near the clean end of the chain. Figure 4 reports reconstruction mean absolute error (MAE) and simulator-based dynamics root mean square error (RMSE) under these ablations. Full-chain likelihood guidance has the largest effect on anchor preservation: averaged over the four corruption patterns, it reduces reconstruction MAE by 47.3% compared with sparse guidance and by 99.1% compared with clean-level-only guidance. This suggests that edit constraints should guide the denoising process throughout the chain rather than only at the final clean sample. Transition-consistency guidance provides a complementary feasibility gain: with full-chain likelihood guidance enabled, adding the transition potential further reduces simulator dynamics RMSE by 17.6% on average, with the clearest

Table 1: Spatial editing results on D4RL locomotion tasks. We report normalized score with standard deviations over 4 random seeds. Best is highlighted in red, second-best in blue.

Policy	Method	Med-Expert			Medium			Med-Replay			Avg.
		HalfCheetah	Hopper	Walker2d	HalfCheetah	Hopper	Walker2d	HalfCheetah	Hopper	Walker2d	
IQL	Original	$92.7_{\pm 2.5}$	$98.7_{\pm 10}$	$112.4_{\pm 0.8}$	$48.5_{\pm 0.3}$	$65.9_{\pm 5.7}$	$81.1_{\pm 1.8}$	$44.1_{\pm 0.5}$	$91.4_{\pm 8.1}$	$80.7_{\pm 7.0}$	79.5
	SER	$88.9_{\pm 2.1}$	$110.4_{\pm 1.6}$	$111.7_{\pm 1.1}$	$49.3_{\pm 0.1}$	$66.6_{\pm 2.4}$	$85.9_{\pm 1.6}$	$46.6_{\pm 0.1}$	$102.4_{\pm 0.5}$	$85.7_{\pm 3.6}$	83.1
	TATU	$94.4_{\pm 0.6}$	$93.4_{\pm 17.8}$	$110.7_{\pm 0.6}$	$48.2_{\pm 0.1}$	$60.3_{\pm 3.6}$	$76.6_{\pm 10.7}$	$44.2_{\pm 0.1}$	$79.6_{\pm 7.6}$	$75.0_{\pm 12.1}$	75.8
	DStitch	$94.4_{\pm 1.4}$	$110.9_{\pm 2.9}$	$111.6_{\pm 0.1}$	$49.4_{\pm 0.14}$	$71.0_{\pm 4.2}$	$83.2_{\pm 2.2}$	$44.7_{\pm 0.1}$	$102.1_{\pm 0.4}$	$86.6_{\pm 2.8}$	83.8
	BITE	$95.7_{\pm 0.5}$	$111.3_{\pm 1.9}$	$113.1_{\pm 0.2}$	$50.3_{\pm 0.4}$	$84.5_{\pm 4.1}$	$90.1_{\pm 2.3}$	$47.0_{\pm 0.4}$	$102.5_{\pm 0.2}$	$88.7_{\pm 3.5}$	87.0
TD3+BC	Original	$94.3_{\pm 0.9}$	$94.8_{\pm 13}$	$109.9_{\pm 0.8}$	$48.4_{\pm 0.1}$	$58.0_{\pm 2.8}$	$81.4_{\pm 2.3}$	$44.5_{\pm 0.2}$	$64.8_{\pm 19.2}$	$82.4_{\pm 5.0}$	75.4
	SER	$86.5_{\pm 8.8}$	$104.0_{\pm 10.8}$	$110.5_{\pm 0.3}$	$48.4_{\pm 0.4}$	$56.4_{\pm 1.3}$	$84.9_{\pm 0.3}$	$45.2_{\pm 0.1}$	$56.8_{\pm 9.9}$	$89.1_{\pm 0.5}$	75.8
	TATU	$89.3_{\pm 3.9}$	$99.0_{\pm 14.9}$	$110.7_{\pm 0.7}$	$48.1_{\pm 0.2}$	$58.3_{\pm 4.8}$	$75.8_{\pm 3.5}$	$44.5_{\pm 0.3}$	$64.1_{\pm 10.5}$	$62.1_{\pm 10.4}$	72.4
	DStitch	$96.0_{\pm 0.5}$	$107.1_{\pm 7.0}$	$110.2_{\pm 0.3}$	$50.4_{\pm 0.5}$	$60.3_{\pm 4.9}$	$83.4_{\pm 1.7}$	$44.7_{\pm 0.3}$	$79.6_{\pm 13.5}$	$89.7_{\pm 4.2}$	80.2
	BITE	$97.6_{\pm 0.8}$	$108.7_{\pm 3.8}$	$111.2_{\pm 0.2}$	$51.3_{\pm 0.3}$	$75.6_{\pm 3.9}$	$90.2_{\pm 1.4}$	$47.1_{\pm 0.3}$	$85.2_{\pm 12.1}$	$90.3_{\pm 5.2}$	84.1
DT	Original	$90.8_{\pm 1.4}$	$109.9_{\pm 1.9}$	$108.1_{\pm 0.3}$	$40.4_{\pm 2.2}$	$61.5_{\pm 2.2}$	$70.5_{\pm 1.6}$	$39.8_{\pm 1.6}$	$83.6_{\pm 3.9}$	$53.3_{\pm 11.2}$	73.1
	DStitch	$92.6_{\pm 0.1}$	$109.4_{\pm 1.9}$	$108.6_{\pm 0.4}$	$44.2_{\pm 0.3}$	$60.5_{\pm 4.3}$	$72.0_{\pm 4.9}$	$41.0_{\pm 0.4}$	$96.1_{\pm 2.0}$	$60.2_{\pm 1.8}$	76.1
	BITE	$93.8_{\pm 0.3}$	$111.3_{\pm 0.8}$	$108.9_{\pm 0.5}$	$44.2_{\pm 0.2}$	$76.5_{\pm 2.5}$	$77.4_{\pm 3.0}$	$42.3_{\pm 0.2}$	$91.0_{\pm 2.2}$	$75.9_{\pm 7.6}$	80.1

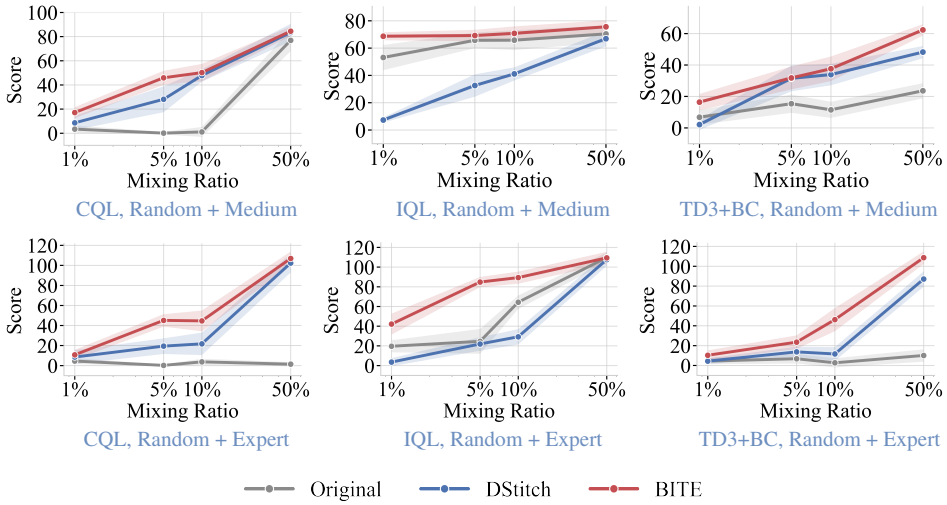


Figure 5: Evaluation on the challenging Walker2d benchmark. The mixing ratio denotes the retained fraction of medium or expert data.

benefit under occlusion. Together, these results show that the lifted likelihood enforces global edit consistency, while the transition potential improves local dynamics plausibility.

6.2 Spatial Trajectory Editing on D4RL Locomotion Tasks

We next evaluate spatial editing as offline RL data augmentation. Each edit keeps two trajectory fragments fixed as anchors and infers a bridge between them. The edited trajectories are then added to the original offline dataset for downstream policy learning.

Standard benchmark. We evaluate on the standard D4RL locomotion benchmark, including HalfCheetah, Hopper, and Walker2d under medium-expert, medium, and medium-replay datasets (Fu et al., 2020). We use IQL (Kostrikov et al., 2022), TD3+BC (Fujimoto & Gu, 2021), and Decision Transformer (DT) (Chen et al., 2021) as downstream learners, following the same setting as in (Li et al., 2024). BITE is compared with the original datasets and three augmentation baselines: SER (Lu et al., 2023), TATU (Zhang et al., 2023), and DStitch (Li et al., 2024). For DT, SER and TATU do not produce trajectory-level data suitable for sequence modeling.

Table 1 shows that BITE delivers the strongest overall performance across downstream learners, achieving the best result in 26 out of 27 task–learner combinations. The gains are most pronounced on the Medium split, where the data contain suboptimal behavior but no expert demonstrations. On this split, BITE improves over DStitch by 11.4% on average, indicating that trajectory composition is especially useful when the offline data lack expert behavior. BITE remains competitive on Medium-Expert and Medium-Replay, where several baselines are already close to performance saturation.

Challenging benchmark. We further test BITE on the mixed-quality Walker2d benchmark (Hong et al., 2023a;b), where random trajectories are mixed with only a small fraction of medium or

Table 2: Temporal editing results on RoboMimic manipulation tasks. We report success rates with standard deviations over 4 random seeds. Red indicates the best result.

Policy	Method	Lift		Can		Square		Avg.
		PH	MH	PH	MH	PH	MH	
BCQ	Original	$95.3_{\pm 1.0}$	$93.4_{\pm 2.1}$	$71.5_{\pm 2.4}$	$55.8_{\pm 4.6}$	$50.3_{\pm 4.2}$	$13.5_{\pm 3.6}$	63.3
	BITE	$97.2_{\pm 0.4}$	$95.8_{\pm 1.0}$	$79.2_{\pm 4.8}$	$60.3_{\pm 3.8}$	$56.5_{\pm 4.6}$	$17.8_{\pm 3.8}$	67.8
CQL	Original	$91.4_{\pm 5.5}$	$60.2_{\pm 32.7}$	$36.7_{\pm 8.3}$	$23.4_{\pm 5.3}$	$5.3_{\pm 2.3}$	$1.3_{\pm 1.0}$	36.4
	BITE	$94.3_{\pm 6.6}$	$80.4_{\pm 13.1}$	$45.3_{\pm 11.8}$	$27.0_{\pm 5.8}$	$13.5_{\pm 7.0}$	$6.7_{\pm 2.2}$	44.5
IQL	Original	$96.3_{\pm 1.5}$	$85.8_{\pm 4.7}$	$73.4_{\pm 5.2}$	$54.8_{\pm 7.3}$	$40.4_{\pm 3.6}$	$13.4_{\pm 1.8}$	60.7
	BITE	$99.2_{\pm 0.5}$	$90.2_{\pm 3.4}$	$82.3_{\pm 6.8}$	$63.4_{\pm 13.7}$	$55.6_{\pm 1.2}$	$16.4_{\pm 2.3}$	67.9
IDQL	Original	$100.0_{\pm 0.0}$	$99.5_{\pm 0.3}$	$97.3_{\pm 3.2}$	$82.6_{\pm 5.8}$	$76.5_{\pm 13.7}$	$58.2_{\pm 2.9}$	85.7
	BITE	$100.0_{\pm 0.0}$	$100.0_{\pm 0.0}$	$100.0_{\pm 0.0}$	$94.5_{\pm 3.7}$	$80.0_{\pm 4.5}$	$66.4_{\pm 2.2}$	90.2

expert data. This setting creates a stronger connectivity challenge: useful trajectory fragments are rare, and the dataset is dominated by low-quality behavior. We train IQL (Kostrikov et al., 2022), TD3+BC (Fujimoto & Gu, 2021), and CQL (Kumar et al., 2020). We exclude DT because we find that it is highly affected by low-quality data and generally fails on this challenging benchmark. We compare BITE with the original mixed datasets and DStitch.

Figure 5 shows that BITE is particularly strong in the mixed-quality Walker2d benchmark. Across the 24 algorithm–mixture–ratio settings, BITE achieves the best score in 23 cases and obtains the best average score for all three downstream learners. Averaged over all settings, BITE improves the score from 35.9 to 56.3 compared with DStitch, yielding a 56.7% relative improvement. The advantage becomes even more pronounced as high-quality trajectories become scarce. At the 10% mixing ratio, BITE improves over DStitch by 82.8% on average. At 5%, the gain increases to 104.0%, meaning that BITE more than doubles the average score of DStitch. In the most challenging 1% setting, BITE reaches nearly $4.8\times$ the average score of DStitch. This is the regime of Proposition 1: when useful fragments are rare, BITE still imposes the bridge constraint at sampling time, whereas an amortized stitcher such as DStitch sees few matched examples of such rare connections during training. More experimental results, including additional controls, comparisons, sensitivity analyses, and a cost profile, are reported in Appendix J.

6.3 Temporal Trajectory Editing on RoboMimic Manipulation Tasks

We finally evaluate temporal editing on RoboMimic manipulation tasks (Mandlekar et al., 2022). Given a source demonstration, BITE anchors selected source states and infers the intermediate states and actions. The edited demonstrations are added to the original dataset and used to train BCQ (Fujimoto et al., 2019), CQL, IQL, and IDQL (Hansen-Estruch et al., 2023).

Table 2 shows that temporal editing consistently improves manipulation performance. Averaged over all policies and tasks, it increases the success rate from 61.5 to 67.6, giving a 9.9% relative improvement. The relative gains are especially clear for CQL and IQL, where BITE improves the average success rate by 22.4% and 11.8%, respectively. Importantly, the improvement also holds for IDQL, which is already a very strong baseline on the original data. Across policies, the gains are concentrated on the more precision-sensitive tasks. BITE improves average success by 11.4% on Can and 20.9% on Square, compared with 4.9% on the already near-saturated Lift task. The gains are also larger on the Multi-Human (MH) split than on the Proficient-Human (PH) split (12.0% vs. 8.2%), suggesting that temporal densification is especially useful when demonstrations contain heterogeneous timing or under-sampled interaction phases.

7 Conclusion

Offline RL is limited by structural deficiencies in fixed datasets, including spatial gaps and temporal sparsity. We introduced BITE, a Bayesian trajectory-editing framework that repairs such deficiencies without additional environment interaction. BITE combines a diffusion trajectory prior with an anchor-consistency likelihood and uses lifted posterior sampling to produce edits that are both constraint-satisfying and data-plausible. Experiments show that BITE improves downstream offline RL performance across multiple learners. Its main limitation is the computational cost of particle-based posterior sampling, which grows with the number of particles; in practice, however, editing is a one-time offline step and the unconditional prior only needs a small backbone, so our profiling in Appendix J.6 finds the cost to be modest. Future work will scale BITE to richer observations, longer horizons, and adaptive edit selection.

Ethics Statement

This work studies trajectory editing for offline reinforcement learning, with the goal of improving the utility of fixed datasets without additional environment interaction. Its potential positive impact is to make offline RL more data-efficient and reduce the need for costly online trial-and-error data collection, especially in domains where collecting new interaction data is expensive or impractical. All experiments are conducted on standard simulation benchmarks, and the proposed method is intended as a research tool for data augmentation rather than a standalone deployment system. As with offline RL methods more broadly, practical use in high-stakes domains should be accompanied by task-specific validation, safety evaluation, and careful monitoring. We do not foresee specific ethical concerns beyond those common to offline RL research.

Reproducibility Statement

The complete BITE algorithm is given in Algorithm 1 (Appendix A). Full proofs of Theorem 1 and Proposition 1, with all assumptions stated explicitly, are in Appendices B and C. Environment, dataset, and anchor-construction details are in Appendices D–F.2; hyperparameters are listed in Appendix H; sensitivity sweeps are in Appendix J.4; and the hardware and profiling protocol are in Appendices I and J.6. Anonymized source code is available at `https://anonymous.4open.science/r/bite-055D`.

AI Use Statement

In this work, we used generative AI tools only for editing the text (grammar, spelling, and word choice). We did not use generative AI tools for research ideation, for writing or verifying code, for generating or analyzing experimental results, or for producing figures, and the remaining required-disclosure tasks are not applicable to this work. We have reviewed all AI-assisted edits and take responsibility for the final content of this work.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Gabriel Cardoso, Yazid Janati El Idrissi, Sylvain Le Corff, and Eric Moulines. Monte carlo guided denoising diffusion models for bayesian linear inverse problems. In *International Conference on Learning Representations*, 2024.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

Zibin Dong, Yifu Yuan, Jianye Hao, Fei Ni, Yi Ma, Pengyi Li, and Yan Zheng. Cleandiffuser: An easy-to-use modularized library for diffusion models in decision making. *Advances in Neural Information Processing Systems*, 37:86899–86926, 2024.

Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062, 2019.

Nico Gürtler, Sebastian Blaes, Pavel Kolev, Felix Widmaier, Manuel Wuthrich, Stefan Bauer, Bernhard Schölkopf, and Georg Martius. Benchmarking offline reinforcement learning on real-robot hardware. In *The Eleventh International Conference on Learning Representations*, 2023.

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.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Zhang-Wei Hong, Pulkit Agrawal, Remi Tachet des Combes, and Romain Laroche. Harnessing mixed offline reinforcement learning datasets via trajectory weighting. In *The Eleventh International Conference on Learning Representations*, 2023a.

Zhang-Wei Hong, Aviral Kumar, Sathwik Karnik, Abhishek Bhandwaldar, Akash Srivastava, Joni Pajarinen, Romain Laroche, Abhishek Gupta, and Pulkit Agrawal. Beyond uniform sampling: Offline reinforcement learning with imbalanced datasets. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in neural information processing systems*, 35:23593–23606, 2022.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Dongsu Lee, Chanin Eom, and Minhae Kwon. Ad4rl: Autonomous driving benchmarks for offline reinforcement learning with value-based dataset. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8239–8245, 2024a.

Jaewoo Lee, Sujin Yun, Taeyoung Yun, and Jinkyoo Park. GTA: Generative trajectory augmentation with guidance for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Guanghe Li, Yixiang Shan, Zhengbang Zhu, Ting Long, and Weinan Zhang. Diffstitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. In *International Conference on Machine Learning*, pp. 28597–28609, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Cong Lu, Philip Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36:46323–46344, 2023.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, pp. 1678–1690. PMLR, 2022.

Andrew Melehy, Jeffrey Feng, Dominic Amara, Vatche G Agopian, and Alex AT Bui. Liver transplant donor-recipient matching with offline reinforcement learning. *npj Digital Medicine*, 2026.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations*, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *International Conference on Machine Learning*, 2025b.

Kajetan Schweighofer, Marius-constantin Dinu, Andreas Radler, Markus Hofmarcher, Vihang Prakash Patil, Angela Bitto-Nemling, Hamid Eghbal-Zadeh, and Sepp Hochreiter. A dataset perspective on offline reinforcement learning. In *Conference on Lifelong Learning Agents*, pp. 470–517, 2022.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Yihao Sun, Jiaji Zhang, Chengxing Jia, Haoxin Lin, Junyin Ye, and Yang Yu. Model-bellman inconsistency for model-based offline reinforcement learning. In *International Conference on Machine Learning*, pp. 33177–33194. PMLR, 2023.

Luhuan Wu, Brian L. Trippe, Christian A. Naesseth, David M. Blei, and John P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Junjie Zhang, Jiafei Lyu, Xiaoteng Ma, Jiangpeng Yan, Jun Yang, Le Wan, and Xiu Li. Uncertainty-driven trajectory truncation for data augmentation in offline reinforcement learning. In *ECAI 2023*, pp. 3018–3025. IOS Press, 2023.

Hanye Zhao, Xiaoshen Han, Zhengbang Zhu, Minghuan Liu, Yong Yu, De-Chuan Zhan, and Weinan Zhang. Dydiff: Long-horizon rollout via dynamics diffusion for offline reinforcement learning. *arXiv preprint arXiv:2405.19189*, 2024.

Qinqing Zheng, Mikael Henaff, Brandon Amos, and Aditya Grover. Semi-supervised offline reinforcement learning with action-free trajectories. In *International conference on machine learning*, pp. 42339–42362, 2023.

A Pseudocode of BITE Algorithm

Algorithm 1 Bayesian Inference for Trajectory Editing (BITE)

Require: offline dataset $\mathcal{D}$, editing instance (Ξ, ϵ), number of particles M, denoising steps L
Ensure: edited trajectory ξ

Prior model learning
Fit the diffusion prior $p(x^{[0]}|\mathcal{D})$ through the reverse transitions $p(x^{[l-1]}|x^{[l]}, \mathcal{D})$.
Fit the conditional reward model $p(\tilde{r}_t|\tilde{s}_t, \tilde{a}_t, \mathcal{D})$.
Fit the transition model $f_{\mathcal{D}}$ and construct the transition-consistency potentials $\phi_{\mathcal{D}}^{[l]}$.
Trajectory editing
Determine the edited horizon H, observation map $\mathcal{Y}$, and editing operator $\mathcal{E}$ from (Ξ, ϵ) according to the editing type.
Set $y^{[0]} = \mathcal{Y}(\Xi)$ and generate auxiliary observations $\{y^{[l]}\}_{l=1}^{L}$ according to (5).
Sample initial particles $\{x^{[L](j)}\}_{j=1}^{M} \sim p(x^{[L]})$.
Compute initial weights $w_L^{(j)} \propto p(y^{[L]}|x^{[L](j)})\phi_{\mathcal{D}}^{[L]}(x^{[L](j)})$, normalize them, and resample the particles.
for $l = L, L-1, \ldots, 1$ **do**
    for $j = 1, \ldots, M$ **do**
        Sample $x^{[l-1](j)} \sim q_{l-1}(\cdot|x^{[l](j)}, y^{[l-1]})$ from the proposal distribution (10).
        Compute the incremental weight $w_{l-1}^{(j)}$ using (11).
    end for
    Normalize $\{w_{l-1}^{(j)}\}_{j=1}^{M}$ and resample the particles.
end for
Sample one trajectory from the final particle set at level 0 to obtain the edited state-action trajectory $\varphi(\xi) = \{(\tilde{s}_t, \tilde{a}_t)\}_{t=0}^{H-1}$.
Generate rewards $\tilde{r}_t \sim p(\cdot|\tilde{s}_t, \tilde{a}_t, \mathcal{D})$ for $t = 0, \ldots, H-1$.
Return $\xi = \{(\tilde{s}_t, \tilde{a}_t, \tilde{r}_t)\}_{t=0}^{H-1}$.

B Proof of Theorem 1

The proof follows the standard consistency argument for Feynman–Kac particle systems (Del Moral et al., 2006), which has also been applied to diffusion-based sequential Monte Carlo samplers such as TDS (Wu et al., 2023), MCGDiff (Cardoso et al., 2024), and FPS (Dou & Song, 2024).

Proof. Throughout the proof, the standard regularity conditions required for sequential Monte Carlo consistency are satisfied: the horizon H and number of diffusion levels L are finite; the Gaussian likelihoods are non-degenerate; the intermediate normalizing constants are finite and strictly positive; the predictive likelihoods m_{l-1} are finite, positive, and bounded; the incremental weights are bounded; and the mutation-weight operators below map bounded continuous functions to bounded continuous functions. These conditions hold naturally for the BITE construction because the reverse diffusion transitions and lifted edit likelihoods are Gaussian, the edit operator is linear, and $0 < \phi_{\mathcal{D}}^{[l]} \leq 1$.

Let Π_l denote the normalized probability measure associated with the intermediate target π_l. Write γ_l for the corresponding unnormalized measure and $Z_l = \gamma_l(1)$ for its normalizing constant, so that $\Pi_l = \gamma_l / Z_l$. After resampling at level l, let

$$\eta_l^M = \frac{1}{M} \sum_{j=1}^{M} \delta_{x^{[l:L](j)}}$$

be the empirical measure of the genealogical suffix particles. Here $x^{[l:L](j)}$ denotes the full suffix path associated with particle j; when a particle is resampled, its whole genealogy is copied. We prove by backward induction that

$$\eta_l^M \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_l, \qquad l = L, L-1, \ldots, 0.$$

Equivalently, for every bounded continuous test function f, $\eta_l^M(f) \to \Pi_l(f)$ in probability (Durrett, 2019).

First consider level L. Define

$$G_L(x^{[L]}) = \phi_{\mathcal{D}}^{[L]}(x^{[L]})p(y^{[L]}|x^{[L]}).$$

BITE samples $x^{[L](j)} \sim p(x^{[L]})$ and assigns weight $G_L(x^{[L](j)})$. For any bounded continuous f, the law of large numbers gives

$$\frac{\frac{1}{M}\sum_{j=1}^M G_L(x^{[L](j)})f(x^{[L](j)})}{\frac{1}{M}\sum_{j=1}^M G_L(x^{[L](j)})} \xrightarrow[M\to\infty]{\mathbb{P}} \frac{\int p(x^{[L]})G_L(x^{[L]})f(x^{[L]})\,dx^{[L]}}{\int p(x^{[L]})G_L(x^{[L]})\,dx^{[L]}} = \Pi_L(f).$$

The denominator is strictly positive by the positivity of the normalizing constant. After multinomial resampling, the resampled empirical average has, conditionally on the weighted particles, the weighted average above as its conditional mean and conditional variance of order M^{-1}. Hence the resampling error vanishes in probability, and therefore

$$\eta_L^M \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_L.$$

Now assume that

$$\eta_l^M \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_l$$

for some $l \in \{1, \dots, L\}$. We prove the same claim for level $l-1$. Given a parent particle $x^{[l:L](j)}$, BITE samples

$$x^{[l-1](j)} \sim q_{l-1}(\cdot|x^{[l](j)}, y^{[l-1]}) = \frac{p(\cdot|x^{[l](j)}, \mathcal{D})p(y^{[l-1]}|\cdot)}{m_{l-1}(x^{[l](j)})},$$

and uses the incremental weight

$$w_{l-1}(x^{[l-1:l](j)}) = \phi_{\mathcal{D}}^{[l-1]}(x^{[l-1](j)})m_{l-1}(x^{[l](j)}).$$

For a bounded continuous test function f on suffixes $x^{[l-1:L]}$, define the mutation-weight operator

$$Q_{l-1}f(x^{[l:L]}) = \int q_{l-1}(u|x^{[l]}, y^{[l-1]})w_{l-1}(u, x^{[l]})f(u, x^{[l:L]})\,du.$$

Also write $Q_{l-1}1$ for the same expression with $f \equiv 1$.

Let

$$\widehat{\gamma}_{l-1}^M(f) = \frac{1}{M}\sum_{j=1}^M w_{l-1}(x^{[l-1:l](j)})f(x^{[l-1:L](j)})$$

be the unnormalized empirical estimate after mutation and before resampling. Conditionally on the level-l particle system, the samples $x^{[l-1](j)}$ are independent, and therefore

$$\mathbb{E}\big[\widehat{\gamma}_{l-1}^M(f)|\eta_l^M\big] = \eta_l^M(Q_{l-1}f).$$

Because the incremental weights are bounded and f is bounded, the conditional variance of $\widehat{\gamma}_{l-1}^M(f)$ is of order M^{-1}. Hence

$$\widehat{\gamma}_{l-1}^M(f) - \eta_l^M(Q_{l-1}f) \xrightarrow[M\to\infty]{\mathbb{P}} 0.$$

By the induction hypothesis and the continuity of $Q_{l-1}f$,

$$\eta_l^M(Q_{l-1}f) \xrightarrow[M\to\infty]{\mathbb{P}} \Pi_l(Q_{l-1}f).$$

The same argument with $f \equiv 1$ gives

$$\widehat{\gamma}_{l-1}^M(1) \xrightarrow[M\to\infty]{\mathbb{P}} \Pi_l(Q_{l-1}1).$$

It remains to identify the limiting ratio. By the definition of the proposal and incremental weight,

$$q_{l-1}(x^{[l-1]}|x^{[l]}, y^{[l-1]})w_{l-1}(x^{[l-1:l]}) = p(x^{[l-1]}|x^{[l]}, \mathcal{D})p(y^{[l-1]}|x^{[l-1]})\phi_{\mathcal{D}}^{[l-1]}(x^{[l-1]}).$$

Thus applying Q_{l-1} to the level-l target exactly appends the next reverse diffusion transition, the level-$(l-1)$ edit likelihood, and the level-$(l-1)$ transition-consistency potential. Therefore

$$\Pi_l(Q_{l-1}f) = \frac{\gamma_{l-1}(f)}{Z_l}, \qquad \Pi_l(Q_{l-1}1) = \frac{Z_{l-1}}{Z_l}.$$

Since $Z_{l-1} > 0$, Slutsky's theorem yields

$$\frac{\widehat{\gamma}^M_{l-1}(f)}{\widehat{\gamma}^M_{l-1}(1)} \xrightarrow[M\to\infty]{\mathbb{P}} \frac{\Pi_l(Q_{l-1}f)}{\Pi_l(Q_{l-1}1)} = \frac{\gamma_{l-1}(f)}{Z_{l-1}} = \Pi_{l-1}(f).$$

This is the convergence of the weighted particle approximation before resampling. The subsequent multinomial resampling step again preserves the limit: conditionally on the weighted particles, the resampled empirical average has the weighted average as its conditional mean and conditional variance of order M^{-1}. Hence

$$\eta^M_{l-1} \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_{l-1}.$$

By backward induction,

$$\eta^M_0 \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_0.$$

Finally, let $\mathrm{pr}_0(x^{[0:L]}) = x^{[0]}$ be the clean-level projection. For every bounded continuous function g on the clean trajectory space,

$$\frac{1}{M}\sum_{j=1}^{M} g(x^{[0](j)}) = \eta^M_0(g\circ \mathrm{pr}_0) \xrightarrow[M\to\infty]{\mathbb{P}} \Pi_0(g\circ \mathrm{pr}_0) = \big(\Pi_0 \circ (x^{[0]})^{-1}\big)(g).$$

This is exactly

$$\frac{1}{M}\sum_{j=1}^{M} \delta_{x^{[0](j)}} \xrightarrow[M\to\infty]{\mathbb{WP}} \Pi_0 \circ (x^{[0]})^{-1}.$$

□

C Proof of Proposition 1

Throughout, the likelihood $p(\Xi|\xi,\epsilon)$ is the same model-free function for all methods, and the edit condition ϵ follows an arbitrary fixed distribution ν that is independent of the trajectory. All densities are assumed to exist with respect to common dominating measures, and $p^\star \ll p_\theta$, which is implied by $D_{\mathrm{KL}}(p^\star\|p_\theta) < \infty$. We write

$$m^\star(\Xi|\epsilon) = \int p(\Xi|\xi,\epsilon)\, p^\star(\xi)\, d\xi, \qquad m_\theta(\Xi|\epsilon) = \int p(\Xi|\xi,\epsilon)\, p_\theta(\xi)\, d\xi,$$

for the induced distributions of edit instances, and

$$p^\star(\xi|\Xi,\epsilon) = \frac{p(\Xi|\xi,\epsilon)\, p^\star(\xi)}{m^\star(\Xi|\epsilon)}, \qquad p_\theta(\xi|\Xi,\epsilon) = \frac{p(\Xi|\xi,\epsilon)\, p_\theta(\xi)}{m_\theta(\Xi|\epsilon)},$$

for the oracle and Bayesian posteriors, defined for $m^\star$-almost every (Ξ,ϵ) (note $m^\star \ll m_\theta$ since $p^\star \ll p_\theta$). Expectations over $(\Xi,\epsilon)\sim m^\star$ mean $\epsilon\sim\nu$ and $\Xi\sim m^\star(\cdot|\epsilon)$.

C.1 Part (i): Bayesian editing does not amplify the prior error

Consider the two joint distributions over (ξ,Ξ,ϵ),

$$P^\star(\xi,\Xi,\epsilon) = \nu(\epsilon)\, p^\star(\xi)\, p(\Xi|\xi,\epsilon), \qquad P_\theta(\xi,\Xi,\epsilon) = \nu(\epsilon)\, p_\theta(\xi)\, p(\Xi|\xi,\epsilon),$$

which share the factors $\nu(\epsilon)$ and $p(\Xi|\xi,\epsilon)$. We compute $D_{\mathrm{KL}}(P^\star\|P_\theta)$ by the chain rule for KL divergence in both factorization orders.

Factorizing ξ first. Since ϵ is independent of ξ and the conditionals $p(\Xi|\xi,\epsilon)$ coincide,

$$D_{\mathrm{KL}}(P^\star\|P_\theta) = D_{\mathrm{KL}}(p^\star\|p_\theta) + \mathbb{E}_{\xi\sim p^\star,\,\epsilon\sim\nu}\big[D_{\mathrm{KL}}\big(p(\cdot|\xi,\epsilon)\,\|\,p(\cdot|\xi,\epsilon)\big)\big] = D_{\mathrm{KL}}(p^\star\|p_\theta).$$

Factorizing (Ξ, ϵ) *first.* The (Ξ, ϵ)-marginals of $P^\star$ and P_θ are $\nu(\epsilon)m^\star(\Xi|\epsilon)$ and $\nu(\epsilon)m_\theta(\Xi|\epsilon)$, and the ξ-conditionals are the two posteriors, so

$$D_{\mathrm{KL}}(P^\star \| P_\theta) = D_{\mathrm{KL}}(m^\star \| m_\theta) + \mathbb{E}_{(\Xi,\epsilon)\sim m^\star}\big[D_{\mathrm{KL}}\big(p^\star(\cdot|\Xi,\epsilon) \,\|\, p_\theta(\cdot|\Xi,\epsilon)\big)\big],$$

where $D_{\mathrm{KL}}(m^\star \| m_\theta) = \mathbb{E}_{\epsilon\sim\nu}[D_{\mathrm{KL}}(m^\star(\cdot|\epsilon)\|m_\theta(\cdot|\epsilon))] \geq 0$ because the ν factor cancels.

Equating the two expressions gives the identity

$$\mathbb{E}_{(\Xi,\epsilon)\sim m^\star}\big[D_{\mathrm{KL}}\big(p^\star(\cdot|\Xi,\epsilon) \,\|\, p_\theta(\cdot|\Xi,\epsilon)\big)\big] = D_{\mathrm{KL}}(p^\star \| p_\theta) - D_{\mathrm{KL}}(m^\star \| m_\theta), \tag{14}$$

and since $D_{\mathrm{KL}}(m^\star \| m_\theta) \geq 0$, Assumption 1 yields the bound $\leq \delta$ in (12). No change of measure appears at any step: the expectation on the left is under the distribution of edit instances induced by the data, and no training edit distribution enters. This proves part (i). □

C.2 Part (ii): Amortized editing can amplify the error by a density ratio

Define the nonnegative error profile

$$A(\Xi, \epsilon) = D_{\mathrm{KL}}\big(p^\star(\cdot|\Xi,\epsilon) \,\|\, p_\phi(\cdot|\Xi,\epsilon)\big) \;\geq 0.$$

Assumption 2 states $\mathbb{E}_{(\Xi,\epsilon)\sim\rho}[A(\Xi,\epsilon)] \leq \delta$. If $m^\star$ (understood as the joint $\nu(\epsilon)m^\star(\Xi|\epsilon)$) is not absolutely continuous with respect to ρ, then $\|dm^\star/d\rho\|_\infty = \infty$ by convention and (13) holds trivially. Otherwise let $r = dm^\star/d\rho$ be the Radon–Nikodym derivative, so that $\|r\|_{L^\infty(\rho)} = \|dm^\star/d\rho\|_\infty$. Then

$$\mathbb{E}_{(\Xi,\epsilon)\sim m^\star}[A(\Xi,\epsilon)] = \mathbb{E}_{(\Xi,\epsilon)\sim\rho}\big[r(\Xi,\epsilon)\,A(\Xi,\epsilon)\big] \leq \|r\|_{L^\infty(\rho)}\,\mathbb{E}_{(\Xi,\epsilon)\sim\rho}[A(\Xi,\epsilon)] \leq \|r\|_{L^\infty(\rho)}\,\delta,$$

which is (13).

Tightness. Fix a measurable edit region B with $\rho(B) > 0$ and consider an amortized editor whose error profile is

$$A(\Xi, \epsilon) = \frac{\delta}{\rho(B)}\,\mathbf{1}_B(\Xi, \epsilon).$$

Such an editor exists: for $(\Xi,\epsilon) \notin B$ set $p_\phi(\cdot|\Xi,\epsilon) = p^\star(\cdot|\Xi,\epsilon)$, and for $(\Xi,\epsilon) \in B$ choose $p_\phi(\cdot|\Xi,\epsilon)$ with $D_{\mathrm{KL}}(p^\star(\cdot|\Xi,\epsilon)\|p_\phi(\cdot|\Xi,\epsilon)) = \delta/\rho(B)$, which is possible whenever the posterior is non-degenerate (e.g., by mixing with a fixed reference density). Then $\mathbb{E}_\rho[A] = \delta$, so Assumption 2 holds with equality, while the conditional error on B is

$$\mathbb{E}_{(\Xi,\epsilon)\sim\rho}\big[A(\Xi,\epsilon) \,\big|\, (\Xi,\epsilon) \in B\big] = \frac{1}{\rho(B)}\int_B A\,d\rho = \frac{\delta}{\rho(B)}.$$

Hence the density-ratio factor cannot be removed in general: on a region of small training mass, the amortized guarantee degrades as $1/\rho(B)$, and if $\rho(B) = 0$ while $m^\star(B) > 0$, Assumption 2 places no constraint on the editor there at all. This proves part (ii). □

D Experiment Setting in Toy Example

D.1 Customized PointMaze Environment

We build a customized MuJoCo PointMaze environment based on `PointMaze_UMaze-v3`, using the following maze layout:

$$\begin{array}{ccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & S & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & R & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & G & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Here, 1 denotes walls, 0 denotes free cells, R is the reset cell, S is an intermediate waypoint, and G is the final goal. Their approximate PointMaze coordinates are

$$R = (0, 0), \qquad S = (2, 2), \qquad G = (-1, -2).$$

The task requires the point mass to move from R around the obstacle, through the upper corridor, and finally to G.

The state and action are

$$s_t = (p_t^x, p_t^y, v_t^x, v_t^y) \in \mathbb{R}^4, \qquad a_t \in \mathbb{R}^2,$$

so each state-action token has dimension $d_\xi = d_s + d_a = 6$.

We collect offline trajectories with a sparse waypoint-following PD controller:

$$a_t = \mathrm{clip}_{[-1,1]^2}\left(K_p(w_k - p_t) - K_d v_t\right),$$

where $p_t = (p_t^x, p_t^y)$, $v_t = (v_t^x, v_t^y)$, $K_p = 5.0$, $K_d = 0.5$. We use sparse waypoints at turns and long-segment midpoints along the route, with small perturbations applied to waypoint locations to induce trajectory diversity.

The final dataset contains 2500 successful trajectories and 745,326 transitions. Episode lengths range from 253 to 340 steps, with mean length 298.13 and median length 298. These trajectories are used to train a trajectory diffusion prior on normalized state-action windows of horizon $H = 60$.

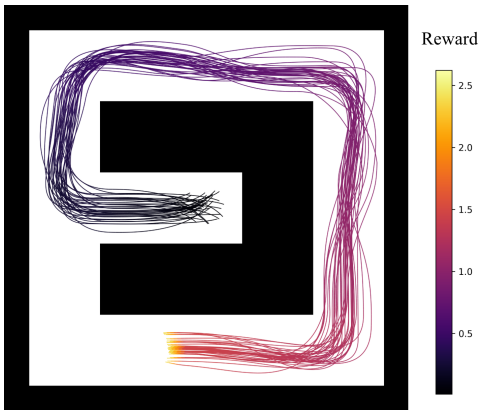


Figure 6: Visualization of 40 successful trajectories collected in the customized PointMaze environment. The trajectories follow the intended route while preserving moderate variation from waypoint perturbations.

The environment uses a dense progress reward along the reference route $R \to S \to G$, together with a step penalty and terminal bonus:

$$r_t = r_{\mathrm{dense}}(p_t) - 0.05 + 2.0 \cdot \mathbb{1}\{p_t \in \mathcal{G}_{\mathrm{final}}\}.$$

This reward is used only to construct and validate the offline dataset; the evaluation is performed on fixed trajectory windows sampled from the collected data.

We evaluate reconstruction on fixed-length PointMaze trajectory segments. Let $\xi = (\xi_0, \ldots, \xi_{H-1})$ denote a trajectory, and let $x = \mathrm{vec}(\varphi(\xi)) \in \mathbb{R}^{d_x}$ be the flattened representation of its state-action sequence. We use $H = 60$, $d_x = H(d_s + d_a)$, with $d_s = 4$ and $d_a = 2$. Each corruption is written as

$$y = \mathcal{E}x + \eta,$$

where $\mathcal{E}$ is a known sample-dependent observation operator. For occlusion, subsampling, and dropout, the synthetic corruption itself is noiseless, so $\eta = 0$. For perturbation, η is Gaussian.

D.2 Corruptions

Occlusion. Occlusion masks a contiguous trajectory segment. We set the occlusion length to $L_{\mathrm{occ}} = 0.3H$, sample $t_0 \sim \mathrm{Unif}\{0, \ldots, H - L_{\mathrm{occ}}\}$, and mask all state-action coordinates in $[t_0, t_0 + L_{\mathrm{occ}})$:

$$m_{t,j} = \mathbb{1}\{t < t_0 \text{ or } t \geq t_0 + L_{\mathrm{occ}}\}, \qquad \mathcal{E}_{\mathrm{occ}} = \mathrm{diag}(\mathrm{vec}(m)), \qquad \eta = 0.$$

Subsampling. Subsampling keeps only sparse temporal observations. We use stride $r_{\mathrm{sub}} = 2$, sample an offset $o \sim \mathrm{Unif}\{0, \ldots, r_{\mathrm{sub}} - 1\}$, and retain frame t iff $t \equiv o \pmod{r_{\mathrm{sub}}}$:

$$m_{t,j} = \mathbb{1}\{t \equiv o \pmod{r_{\mathrm{sub}}}\}, \qquad \mathcal{E}_{\mathrm{sub}} = \mathrm{diag}(\mathrm{vec}(m)), \qquad \eta = 0.$$

Dropout. Dropout removes scattered state observations. For each time step, we sample $b_t \sim \text{Bernoulli}(1 - p_{\text{drop}})$ with $p_{\text{drop}} = 0.6$. The mask is applied to state coordinates, while actions remain observed:

$$m_{t,j} = b_t \ (j < d_s), \qquad m_{t,j} = 1 \ (j \geq d_s), \qquad \mathcal{E}_{\text{drop}} = \text{diag}(\text{vec}(m)), \qquad \eta = 0.$$

Perturbation. Perturbation adds noise to the observed trajectory. All time steps are observed, but state coordinates are perturbed by Gaussian noise in normalized space:

$$\mathcal{E}_{\text{pert}} = I, \qquad \eta_{t,j} \sim \mathcal{N}(0, \sigma^2_{\text{pert}}), \quad \sigma_{\text{pert}} = 0.2, \quad j < d_s,$$

and $\eta_{t,j} = 0$ for action coordinates $j \geq d_s$.

D.3 Evaluation Metric

Trajectory reconstruction error. We report full-trajectory MAE over normalized state-action representations for N samples:

$$\text{MAE}_{\text{traj}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{d_x} \left\| x^{(i)} - \hat{x}^{(i)} \right\|_1 .$$

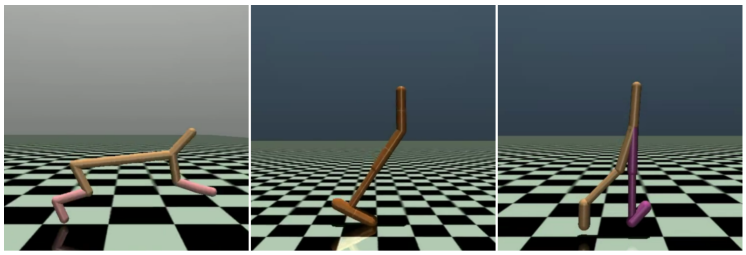
Simulator dynamics consistency. We also report a simulator-based one-step RMSE. Given raw state $s_t^{(i)}$ and reconstructed action $\hat{a}_t^{(i)}$, we compute $\tilde{s}_{t+1}^{(i)} = f_{\text{sim}}\left(s_t^{(i)}, \hat{a}_t^{(i)}\right)$, and report

$$\text{RMSE}_{\text{dyn}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \frac{1}{(H-1)d_s} \sum_{t=0}^{H-2} \left\| \tilde{s}_{t+1}^{(i)} - s_{t+1}^{(i)} \right\|_2^2}.$$

E Experiment Setting in Spatial Trajectory Editing

E.1 Locomotion Environment

For spatial trajectory editing, we evaluate BITE on standard `D4RL` MuJoCo locomotion tasks, including HalfCheetah, Hopper, and Walker2D. These environments test whether edited trajectories can improve offline RL performance under continuous-control dynamics.



Halfcheetah Hopper Walker2D

Figure 7: `D4RL` locomotion tasks used for spatial editing.

E.2 Source Anchor Construction

Spatial source anchors are instantiated under the occlusion setting defined above: the prefix and suffix are observed anchor fragments, while the middle interval is masked for completion. In contrast to the toy occlusion setting, the two anchor fragments are drawn from different trajectories, forming the source content for spatial editing.

Specifically, we first rank trajectories by return-to-go and build two candidate pools. The prefix pool consists of the top ρ_{pre} fraction of trajectories, from which prefixes are sampled. The suffix pool consists of the top ρ_{suf} fraction of high-return trajectories, from which candidate suffixes are

retrieved. We compare the prefix tail with each candidate suffix head using cosine similarity and keep pairs whose similarity exceeds η_{sim}. Following prior practice (Li et al., 2024), we estimate the bridge length for each retained pair.

Bridge-length procedure. For a retained (prefix, suffix) pair, we roll out the prefix with the learned transition model $f_{\mathcal{D}}$ and locate the rollout position whose state is most similar to the head of the candidate suffix; the number of rollout steps to that position determines the gap Δ. Given the editing horizon H, the gap is centered between the two anchor fragments, i.e., the prefix anchor occupies the first $\lfloor (H-\Delta)/2 \rfloor$ steps and the suffix anchor the last $\lceil (H-\Delta)/2 \rceil$ steps, so that $T_a + \Delta + T_b = H$. The prefix fragment, suffix fragment, and estimated bridge length together define the source content for spatial editing.

For each task, we construct an edited transition set matching the size of the original offline dataset, resulting in a 1:1 augmented-to-original data ratio. During policy optimization, each mini-batch is assembled by drawing a fraction λ_{aug} from the augmented set and the remaining samples from the original dataset. We set $\lambda_{\text{aug}} = 0.7$ across all spatial editing experiments.

F Experiment Setting in Temporal Trajectory Editing

F.1 Manipulation Environment

For temporal trajectory editing, we evaluate BITE on `RoboMimic` manipulation tasks, including Lift, Can, and Square. These environments require precise object interaction and are used to evaluate whether temporal trajectory refinement improves offline manipulation policies.

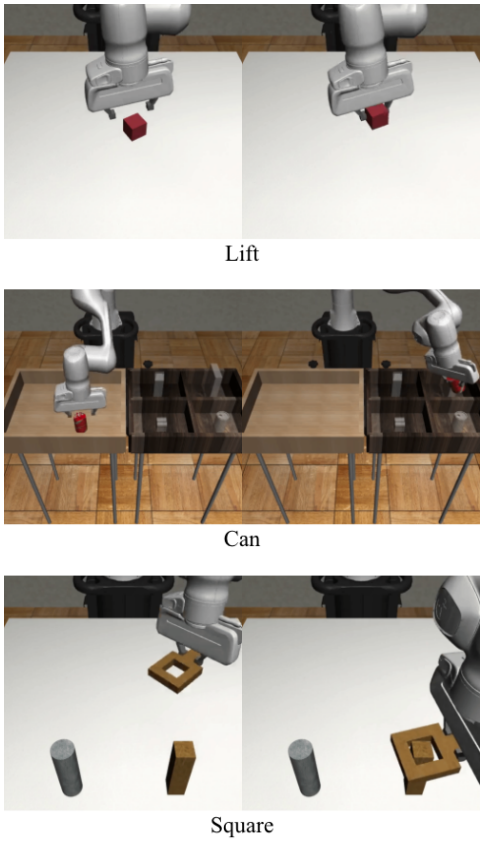


Figure 8: `RoboMimic` manipulation tasks used for temporal editing.

F.2 Source Anchor Construction

Temporal source anchors are instantiated under the subsampling setting defined above: sparse temporal observations are preserved as source states, while the intermediate frames are completed by the diffusion model. Instead of uniform subsampling, we identify temporally uncertain segments using action-distribution entropy.

Specifically, we train a diffusion-BC proxy policy $\pi_\theta(a|o)$ on the original demonstrations to approximate the conditional action distribution. For each timestep t, we draw N_{cand} candidate action samples $\{a_t^{(n)}\}_{n=1}^{N_{\text{cand}}}$ from $a_t^{(n)} \sim \pi_\theta(\cdot|o_t)$ and construct a sample-centered Gaussian mixture density, $\hat{p}(a|o_t) = \frac{1}{N_{\text{cand}}} \sum_{n=1}^{N_{\text{cand}}} \mathcal{N}\Big(a;\, a_t^{(n)},\, h^2 I\Big)$, where each sampled action defines one mixture component and N_{cand} is the number of candidate actions sampled for each observation. We then estimate the conditional action entropy by

$$\hat{H}_t = -\frac{1}{N_{\text{cand}}} \sum_{n=1}^{N_{\text{cand}}} \log \hat{p}\Big(a_t^{(n)}|o_t\Big)\,.$$

The entropy curve is smoothed and robust-normalized within each demonstration, so that τ_{ent} is a relative rather than absolute threshold. Timesteps with normalized entropy at least τ_{ent} are selected and merged into temporal editing segments, since higher entropy indicates greater local action uncertainty (this is the "action chaos" of Figure 1; see Appendix J.1). These segments are replayed with slowdown rate ρ_{slow} to produce temporally expanded source content for temporal editing. We generate one slowed-down trajectory for each original `RoboMimic` demonstration.

G Visualization of Bayesian Editing and Amortized Editing

Figure 9 visualizes spatial stitching examples on `Hopper` and `Walker2d`, where two trajectory fragments are used as anchors and the editor fills the missing middle segment. We plot `Hopper` torso height and `Walker2d` torso angle, as these coordinates clearly expose local discontinuities.

The comparison highlights the difference between amortized editing and Bayesian editing. DStitch amortizes the stitching constraint into a conditional generator trained on edit instances, and can therefore satisfy the coarse anchor request while still producing local artifacts, such as zigzagging torso height in `Hopper` and an abrupt torso-angle change in `Walker2d`. In contrast, BITE learns an unconditional trajectory prior and imposes each concrete edit request at sampling time through an anchor-consistency likelihood, together with transition-consistency guidance. This instance-wise Bayesian conditioning yields smoother and more dynamically plausible bridges. Note that BITE formulates anchor consistency as a Gaussian likelihood rather than a hard equality constraint, the anchors act as soft constraints. Thus, BITE does not necessarily overlap exactly with the anchor fragments, but instead trades off anchor matching, prior plausibility, and local dynamics feasibility.

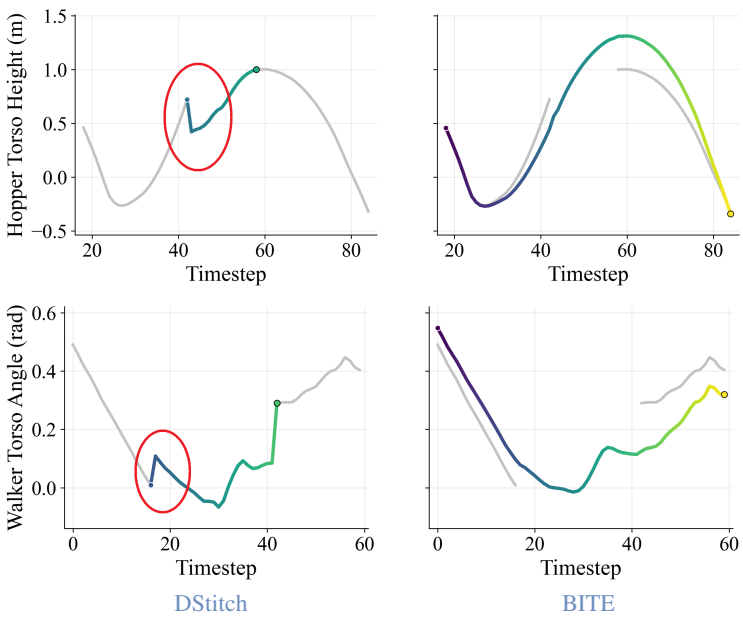


Figure 9: Comparison between amortized editing (**Left**) and Bayesian editing (**Right**).

H Hyperparameters

I Computing Resources

We used local GPU servers for all experiments. Spatial editing experiments on `D4RL` locomotion tasks were conducted on a four-GPU NVIDIA RTX 5090 server, and temporal editing experiments on `RoboMimic` manipulation tasks were conducted on a two-GPU NVIDIA RTX 3090 server.

J Additional Experiments

This appendix collects the experiments that complement Section 6. Most ablations are run on `hopper-medium-v2` with the hyperparameters of Table 3, varying one quantity at a time, and we report the reconstruction error (mean absolute error of the inferred segment against the held-out ground truth), the dynamics error (one-step mean absolute error of the edited trajectory under the true simulator, not under the learned transition model), and the normalized score of IQL trained for 200k steps on the augmented dataset, with mean $\pm$ standard deviation over seeds.

J.1 Operational Definitions of the Two Deficiencies

Figure 1 is plotted from real offline data. In the spatial panel, Traj. 1 and Traj. 2 are trajectory segments from the `D4RL` `Walker2d` dataset and the bridged trajectory is the BITE edit connecting them; the curve underneath plots a stability score against the horizontal position of the walker, for the original fragment, which loses stability and falls, and for the edited trajectory. In the temporal panel, the upper row shows frames of a `RoboMimic` demonstration and the lower row shows the BITE edit, in which the two highlighted phases (grasping and insertion) are slowed down; the curve underneath plots an action-chaos score against task progress.

The stability score adapts the Gym reward decomposition by removing the task-progress term. For `Walker2d` the reward is $r_{\text{forward}} + r_{\text{healthy}} - c_{\text{ctrl}}$; dropping r_{forward} leaves $r_{\text{healthy}} - c_{\text{ctrl}}$, which we normalize per environment, and we call a transition low-stability when its score falls below the normalized threshold. The action-chaos score is the conditional action entropy $\hat{H}_t$ under the diffusion-BC proxy of Appendix F.2, robust-normalized within each demonstration; action-chaos segments are contiguous runs above the threshold τ_{ent}.

Measured this way, low-stability transitions make up 34% of `Walker2d` Medium, 23% of Medium-Replay, and 12% of Medium-Expert. They are most common on the Medium split, which is also

Table 3: Hyperparameters for Spatial Editing and Temporal Editing.

Hyperparameter / Symbol	Value
A. Shared Configuration	
Diffusion backbone	DiT1d (Dong et al., 2024)
Diffusion prior training objective	DDPM (Ho et al., 2020)
Diffusion Sampling steps, L	100
Likelihood noise variance, σ	0.05
Transition consistency guidance horizon, L_{tc}	50
Particle population size, M	20
Transition model type	Ensemble dynamics (Sun et al., 2023)
Number of ensemble models	7
Optimizer	AdamW (Loshchilov & Hutter, 2017)
Learning rate	3×10^{-4}
B. Spatial Editing (`D4RL`)	
Horizon for standard benchmark, H	Hopper: 80; HalfCheetah: 40; Walker2d: 60
Horizon for challenge benchmark, H	18
Total gradient steps	250,000
Diffusion batch size	128
Diffusion embedding dimension	64
Diffusion model dimension	256
Transformer attention heads	8
Transformer depth	6
Ratio of prefix trajectories, ρ_{pre}	0.3
Ratio of suffix trajectories, ρ_{suf}	0.03
Similarity threshold, η_{sim}	0.90
Augmented data portion, λ_{aug}	0.7
C. Temporal Editing (`RoboMimic`)	
Horizon, H	80
Total gradient steps	1,000,000
Diffusion batch size	32
Diffusion embedding dimension	128
Diffusion model dimension	512
Transformer attention heads	16
Transformer depth	12
Candidate action sample size, N_{cand}	32
Slowdown rate for selected segments, ρ_{slow}	2×

where spatial editing helps most in Table 1. Action-chaos segments cover 11% of timesteps on the `RoboMimic` PH datasets and 16% on MH, in line with the larger gains of temporal editing on MH in Table 2. The two deficiencies are thus measurable properties of the datasets rather than assumptions.

J.2 Long-Horizon Stitching and Other Augmentation Methods

`D4RL` locomotion data are relatively clean and short-horizon, so we also evaluate spatial editing on `OGBench` (Park et al., 2025a) `antmaze-large-stitch-v0`, whose data consist of short local fragments that must be composed into long-range behavior, and `antmaze-large-explore-v0`, whose data consist of random-direction exploration with sparse useful transitions. Edits are constructed as in Appendix E.2, and we train the goal-conditioned reference learners GCIQL and GCIVL as well as IQL and FQL (Park et al., 2025b) on the original and augmented datasets. BITE improves all eight dataset–learner combinations (Table 4), including the strong FQL learner, so the benefit carries over to sparse, fragmented, long-horizon data and does not depend on a particular learning objective.

We also compare with two diffusion-based augmentation methods that do not edit trajectories explicitly, GTA (Lee et al., 2024b) and DyDiff (Zhao et al., 2024), on the `D4RL` tasks they share with us, averaging over HalfCheetah, Hopper, and Walker2d for each dataset type. BITE is slightly

stronger than GTA alone and better than DyDiff on every dataset type (Tables 5 and 6). GTA and BITE are complementary, since GTA biases generation toward high-return regions while BITE repairs structural deficiencies, and applying both improves over BITE in all six settings.

Table 4: Success rate (%) on `OGBench antmaze-large` datasets, mean ± std over seeds. Best is highlighted in red.

Dataset	Data	GCIQL	GCIVL	IQL	FQL
Stitch	Original	$7.0_{\pm 2.0}$	$18.0_{\pm 2.0}$	$28.0_{\pm 6.0}$	$91.0_{\pm 4.0}$
	BITE	$8.0_{\pm 2.0}$	$22.0_{\pm 3.0}$	$36.0_{\pm 7.0}$	$97.0_{\pm 2.0}$
Explore	Original	$0.0_{\pm 0.0}$	$10.0_{\pm 3.0}$	$7.0_{\pm 2.0}$	$86.0_{\pm 7.0}$
	BITE	$6.0_{\pm 4.0}$	$14.0_{\pm 3.0}$	$12.0_{\pm 4.0}$	$95.0_{\pm 3.0}$

Table 5: Normalized score on `D4RL`, averaged over the three locomotion environments: GTA, BITE, and their combination. Best is highlighted in red, second-best in blue.

Policy	Dataset	GTA	BITE	GTA+BITE
TD3+BC	Medium	71.4	72.4	75.3
	Med-Replay	77.7	74.2	79.4
	Med-Expert	104.8	105.8	108.0
IQL	Medium	72.2	75.0	77.1
	Med-Replay	80.8	79.4	83.2
	Med-Expert	105.3	106.7	109.4
Avg.		85.4	85.6	88.7

Table 6: Normalized score under TD3+BC, averaged over the three locomotion environments: DyDiff versus BITE. Best is highlighted in red.

Dataset	DyDiff	BITE
Medium	69.2	72.4
Med-Replay	70.2	74.2
Med-Expert	99.7	105.8
Avg.	79.7	84.1

J.3 Isolating the Source of the Gains

Since BITE adds trajectories to the dataset, part of the improvement over the unaugmented baseline could simply come from having more data. To check this, we follow the `D4RL` collection recipe, restore the Hopper-Medium behavior policy, and roll it out to produce a same-policy augmentation set of the same size as the BITE-edited set. Table 7 shows that this extra data barely moves the downstream score (IQL $65.9 \rightarrow 68.3$, TD3+BC $58.0 \rightarrow 58.6$), whereas BITE reaches 84.5 and 75.6 under the same budget. The gains therefore come from what is added rather than how much, and the informative comparison for the editing procedure is the one against DStitch.

A second possible confound is the learned reward model used to label the edits (Eq. 2), since edited state-action pairs may lie near the edge of its training support. We relabel the same BITE-edited trajectories with true simulator rewards, changing nothing else. Table 8 shows that the reward source has little effect for IQL and for the return-conditioned DT, with averages within about one point and no consistent direction, so the improvement comes from the edited state-action structure rather than from reward relabeling. We use simulator relabeling rather than omitting rewards because IQL and DT could not be trained otherwise; joint next-state/reward prediction is also standard in model-based offline RL, e.g., MOBILE (Sun et al., 2023).

Table 7: Matched-budget control on `hopper-medium-v2`. All augmented datasets have the same size. Best is highlighted in red, second-best in blue.

Data	Size	IQL	TD3+BC
Original	1×	$65.9_{\pm 5.7}$	$58.0_{\pm 2.8}$
Original + same-policy rollouts	2×	$68.3_{\pm 5.5}$	$58.6_{\pm 3.6}$
Original + DStitch	2×	$71.0_{\pm 4.2}$	$60.3_{\pm 4.9}$
Original + BITE	2×	$84.5_{\pm 4.1}$	$75.6_{\pm 3.9}$

Table 8: Reward source on Hopper: the same BITE-edited trajectories labeled with the learned reward model or with the true simulator reward.

Policy	Reward	Med-Expert	Medium	Med-Replay	Avg.
IQL	Simulator	$108.6_{\pm 3.1}$	$85.0_{\pm 4.7}$	$101.2_{\pm 0.3}$	98.3
	Learned	$111.3_{\pm 1.9}$	$84.5_{\pm 4.1}$	$102.5_{\pm 0.2}$	99.4
DT	Simulator	$111.4_{\pm 0.2}$	$79.8_{\pm 2.4}$	$88.4_{\pm 1.7}$	93.2
	Learned	$111.3_{\pm 0.8}$	$76.5_{\pm 2.5}$	$91.0_{\pm 2.2}$	92.9

J.4 Role of the Transition-Consistency Potential

The diffusion prior is trained on dynamically valid trajectories, so one may ask whether the potential $\phi_{\mathcal{D}}$ adds anything, and whether relying on a learned one-step model risks pushing edits toward that model's own artifacts. Tables 11 and 12 sweep the strength λ on Hopper-Medium, including $\lambda = 0$. Because the dynamics error is measured with the true simulator rather than with the model that defines the potential, the drop from 8.5×10^{-3} to 6.2×10^{-3} shows that the potential supplies feasibility information that the diffusion trajectory alone does not reliably provide; the IQL score rises from 70.0 to 84.0 and is insensitive across $\lambda \in [0.1, 1.0]$. On `RoboMimic` Can (Table 9, success averaged over IQL and IDQL), moderate guidance improves both PH and MH, with the larger gain on the more heterogeneous MH demonstrations, while $\lambda = 0.5$ degrades both, indicating over-guidance.

As for model artifacts, the transition model is used only as a one-step consistency signal and is never rolled out, and on the same generated Hopper pairs its next-state estimate is far closer to a true simulator step (dynamics error 0.33×10^{-3}) than the diffusion trajectory itself (8.49×10^{-3}). Table 10 applies the guidance to diffusion priors saved at three training checkpoints: it reduces the true-simulator dynamics error at every checkpoint, with larger corrections for less accurate priors, so the imperfect model acts as a useful local correction rather than a source of bias. Finally, the L_{tc} blocks of Tables 11 and 12 vary the number of denoising levels at which the potential is applied. Extending the guidance improves all three metrics up to $L_{\mathrm{tc}} = 50$, whereas applying it through the full chain adds little generation quality and lowers the IQL score, since transition residuals carry little information on very noisy samples.

Table 9: Sweep of λ on `RoboMimic` Can (success rate averaged over IQL and IDQL).

Dataset	$\lambda = 0$	0.05	0.1	0.5
Can-PH	89.7	91.2	90.8	85.3
Can-MH	72.6	79.0	79.4	75.0

Table 10: Dynamics error ($\times 10^{-3}$) of edited Hopper trajectories with and without guidance, for priors at different training checkpoints.

Checkpoint	No guidance	Guided	Reduction
Early	24.0	11.0	54.0%
Middle	13.9	9.23	33.7%
Final	8.49	6.22	26.8%

Table 11: Generation quality of the edits on Hopper-Medium under the four sweeps: reconstruction error and dynamics error ($\times 10^{-3}$). Each sweep varies one quantity with the others held fixed.

λ	0	0.02	0.05	0.1	0.2	0.5	1.0
Reconstruction error	0.204	0.174	0.169	0.164	0.164	0.164	0.164
Dynamics error	8.49	8.22	6.45	6.22	6.22	6.22	6.12
L_{tc}	0	5	10	25	50	100	
Reconstruction error	0.187	0.174	0.168	0.164	0.164	0.163	
Dynamics error	7.05	6.92	6.53	6.22	6.21	6.20	
M	1	2	4	8	16	20	32
Reconstruction error	0.229	0.226	0.207	0.183	0.168	0.154	0.150
Dynamics error	8.02	7.94	7.75	7.05	6.60	6.22	6.76
σ	0.01	0.02	0.05	0.1	0.2	0.5	1.0
Reconstruction error	0.162	0.158	0.164	0.174	0.181	0.219	0.298
Dynamics error	7.09	6.87	6.22	6.63	6.80	6.66	6.38

Table 12: Downstream IQL score on Hopper-Medium under the four sweeps, mean ± std over seeds.

λ	0	0.02	0.05	0.1	0.2	0.5	1.0
IQL score	$70.0_{\pm 14.8}$	$67.2_{\pm 21.2}$	$74.5_{\pm 4.5}$	$84.0_{\pm 3.1}$	$80.5_{\pm 8.3}$	$82.0_{\pm 10.7}$	$83.7_{\pm 1.3}$
L_{tc}	0	5	10	25	50	100	
IQL score	$64.0_{\pm 14.8}$	$71.6_{\pm 10.2}$	$76.4_{\pm 4.5}$	$81.7_{\pm 6.1}$	$85.7_{\pm 5.6}$	$72.4_{\pm 8.0}$	
M	1	2	4	8	16	20	32
IQL score	$71.5_{\pm 3.0}$	$69.6_{\pm 7.9}$	$71.7_{\pm 12.2}$	$79.8_{\pm 6.4}$	$77.5_{\pm 6.3}$	$84.5_{\pm 4.1}$	$80.7_{\pm 14.3}$
σ	0.01	0.02	0.05	0.1	0.2	0.5	1.0
IQL score	$80.3_{\pm 4.9}$	$82.7_{\pm 7.4}$	$84.5_{\pm 4.5}$	$72.8_{\pm 5.0}$	$64.7_{\pm 10.3}$	$51.7_{\pm 5.8}$	$58.5_{\pm 12.6}$

J.5 Particle Count and Anchor Noise

Theorem 1 is asymptotic in the particle count M, while the experiments use $M = 20$. The M blocks of Tables 11 and 12 sweep M on Hopper-Medium: $M = 20$ gives the lowest dynamics error and the highest IQL score, and $M = 32$ improves only the reconstruction error. At the final resampling stage the normalized effective sample size is 76%, i.e., 15.2 effective particles out of 20, so the sampler is far from degenerate. We regard $M = 20$ as a practical operating point rather than an approximation of the $M \to \infty$ limit; FPS (Dou & Song, 2024) reports a similar observation that a small population suffices in guided linear-Gaussian settings.

The Gaussian likelihood in Eq. 4 turns the exact alignment of Section 2 into a soft constraint that approaches the hard one as $\sigma \to 0$. The σ blocks of the same tables sweep σ: the anchor-state error grows monotonically, from 0.0052 at $\sigma = 0.01$ to 0.1874 at $\sigma = 1.0$, and the default $\sigma = 0.05$ lies in the small-noise regime with the lowest dynamics error and the highest IQL score. Since the edits are used as augmentation and the goal is downstream performance rather than exact constraint satisfaction, this small relaxation is acceptable in practice.

J.6 Computational Cost

BITE runs an M-particle sampler over L denoising steps per edit, so we profiled it against a single denoising pass with the same backbone and against the amortized DStitch editor, on one NVIDIA RTX 5090 with CUDA-synchronized wall-clock medians over three runs after one warm-up (Table 13). At the manuscript setting, particle inference adds 18.3% latency and 32.4% peak memory over vanilla denoising, yet BITE is still $2.21\times$ faster and $3.60\times$ lighter than DStitch per edit. This is a consequence of the Bayesian design: because each edit is imposed at sampling time through the likelihood, the diffusion model only has to learn the unconditional prior and an 8M-parameter backbone suffices, whereas the amortized editor absorbs the edit conditioning into a much larger conditional network (60.7M parameters), and the smaller backbone more than offsets the $M = 20$ particles, which are batched on the GPU. DStitch does retain higher throughput at a saturated batch size of 64 (78.6 versus 12.0 trajectories per second). Cost grows linearly with the number of denoising steps ($0.049/0.097/0.192$ s at $L = 25/50/100$, with nearly constant memory) and mildly with the horizon (latency $0.192 \to 0.269$ s and memory $88.7 \to 121.8$ MiB from $H = 100$ to 200). End-to-end, Hopper-Medium generated 2.00M edited transitions in 104 minutes, and the nine `D4RL` production runs took between 16.4 and 207.5 minutes each. As editing is a one-time offline preprocessing step, this cost affects neither policy training nor deployment.

Table 13: Per-edit cost at $L = 100$, $H = 100$ on one RTX 5090 (CUDA-synchronized medians over three runs).

Method	M	**Time per edit (s)**	**Peak GPU memory (MiB)**	**Parameters**
Vanilla denoising	1	0.164	67.0	8.0M
BITE	20	0.194	88.7	8.0M
DStitch	1	0.429	319.7	60.7M