

ASTRO: Adaptive Stitching via Dynamics-Guided Trajectory Rollouts

Hang Yu¹, Di Zhang¹, Qiwei Du², Yanping Zhao¹, Hai Zhang¹
 Guang Chen¹, Eduardo E. Veas³, Junqiao Zhao^{1*}

¹Tongji University, ²University at Buffalo, ³Graz University of Technology

Abstract

Offline reinforcement learning (RL) enables agents to learn optimal policies from pre-collected datasets. However, datasets containing suboptimal and fragmented trajectories present challenges for reward propagation, resulting in inaccurate value estimation and degraded policy performance. While trajectory stitching via generative models offers a promising solution, existing augmentation methods frequently produce trajectories that are either confined to the support of the behavior policy or violate the underlying dynamics, thereby limiting their effectiveness for policy improvement. We propose ASTRO, a data augmentation framework that generates distributionally novel and dynamics-consistent trajectories for offline RL. ASTRO first learns a temporal-distance representation to identify distinct and reachable stitch targets. We then employ a dynamics-guided stitch planner that adaptively generates connecting action sequences via Rollout Deviation Feedback, defined as the gap between target state sequence and the actual arrived state sequence by executing predicted actions, to improve trajectory stitching's feasibility and reachability. This approach facilitates effective augmentation through stitching and ultimately enhances policy learning. ASTRO outperforms prior offline RL augmentation methods across various algorithms, achieving notable performance gain on the challenging OGBench suite and demonstrating consistent improvements on standard offline RL benchmarks such as D4RL.

1 Introduction

Offline reinforcement learning (RL) enables agents to acquire decision-making capabilities from pre-collected datasets, thereby avoiding the expense and safety risks associated with direct environment interaction (Levine et al. 2020; Agarwal, Schuurmans, and Norouzi 2020; Fujimoto and Gu 2021; Janner, Li, and Levine 2021; Kidambi et al. 2020). However, in the absence of online exploration, offline RL faces two persistent challenges: distributional shift and value-function overestimation (Levine et al. 2020; Fujimoto and Gu 2021; Kostrikov, Nair, and Levine 2021). To address these issues, most methods cast offline RL as a constrained optimization problem: maximizing expected returns while restricting the policy to remain within the dataset's state-action distribution (Fujimoto and Gu 2021; Kostrikov, Nair, and Levine 2021).

*Corresponding at zhaojunqiao@tongji.edu.cn

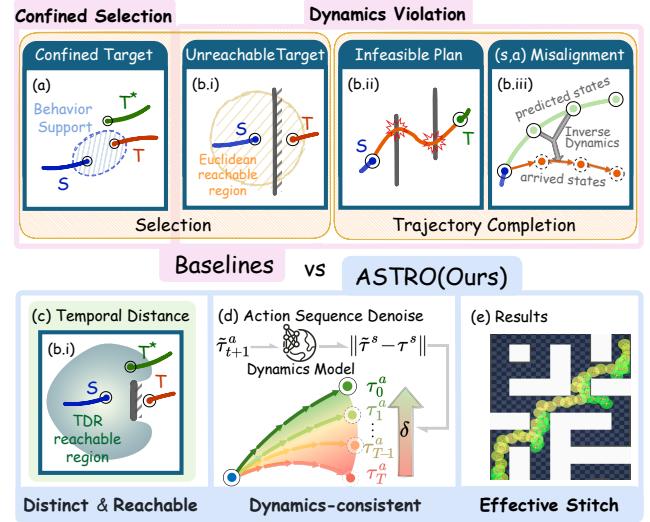


Figure 1: **Comparison of trajectory stitching approaches:** (a) Existing methods suffer from *Confined Target Selection* within behavior policy support; (b) *Dynamics Violation* manifests as: i. Unreachable targets within fixed timesteps behind walls, ii. Infeasible Planning without explicit dynamics modeling, iii. Action-State Misalignment from sub-optimal inverse dynamics; **ASTRO overcomes these via:** (c) Temporal Distance Representation for distinct, reachable target selection beyond behavior support, and (d) Dynamics-Guided Stitching with action sequence denoising using rollout deviation feedback; (e) Resulting in *dynamics-consistent* augmented trajectories that enable effective policy learning.

Nevertheless, when datasets consist of suboptimal and temporally fragmented trajectories, reward signals cannot propagate across trajectory boundaries. This impairs value estimation, disrupts long-horizon consistency, and ultimately degrades policy performance.

A promising solution is trajectory stitching, which augments the dataset by synthesizing new trajectories through bridging desirable sub-trajectories (Li et al. 2024). Prior works (Li et al. 2024) typically identify source-target stitch pairs by generating candidate states from source trajectories and using Euclidean distance metrics to determine viable targets. Synthetic trajectories are then completed using

state-based planners and inverse dynamics models, thereby enriching the data for policy learning.

However, these techniques often underperform in environments with complex dynamics or multi-modal behavior policies (Levine et al. 2020). As illustrated in Fig. 1, stitched trajectories suffer from two major issues: **(a) Confined Target Selection:** Existing approaches constrain stitching to the support of behavior policy by relying on behavior-cloning rollouts to pre-generate candidate targets, limiting novelty and potential policy improvement. **(b) Dynamics Violation:** This manifests in three aspects: *i. Infeasible Target Selection:* Euclidean proximity fails to reflect temporal or semantic feasibility in high-dimensional state space. It may select spatially close yet unreachable states within fixed timesteps (e.g., behind obstacles). *ii. Infeasible Planning:* Previous completion methods entangle policy and dynamics modeling, using state-based planners without explicit dynamics modeling, often producing infeasible plans. *iii. Action-State Misalignment:* Noisy state prediction and suboptimal inverse dynamics lead to inaccurate actions, whose errors compound over long horizons and cause misalignment between planned state and action sequence.

These limitations underscore our key insight: effective trajectory stitching requires breaking through the constraints of behavior policies and establishing explicit alignment between planning decisions and the environment’s underlying dynamics. We propose ASTRO (Adaptive Stitching via dynamics-guided Trajectory Roll-Outs), a model-based data augmentation framework designed to generate distributionally novel and dynamics-consistent stitch trajectories for offline RL learning. ASTRO resolves the aforementioned key limitations as follows: **(1) Stitch Target Selection in Temporal-Distance-Space:** Instead of relying on pre-generated rollouts and naive distance metrics, ASTRO performs stitch target selection via *Temporal Distance Representation (TDR)*, identifying distinct and reachable subtrajectories beyond the behavior distribution. **(2) Decoupled Planning and Explicit Dynamics Modeling:** Instead of direct state-based completion, ASTRO explicitly separates planning from dynamics modeling, employing a planner to propose action sequences and a long-horizon dynamics model for valid and accurate rollouts. **(3) Dynamics-Guided Planning via Rollout Deviation Feedback:** ASTRO utilizes dynamics-guided stitch planner that adaptively generates connecting action sequences via *rollout deviation feedback* (i.e. the gap between target states and the actual reached states) to regularize training and enable adaptive inference, thereby ensures stitching feasibility and further improves target reachability.

To our knowledge, ASTRO is the first trajectory stitching method to achieve substantial performance gains on OGBench, a challenging benchmark with complex dynamics and multi-modal behavior policies. While prior methods yield only marginal improvements, ASTRO improves average task performance by 32.7% (+9.68) across multiple offline RL algorithms. It also provides consistent improvements on standard benchmarks such as D4RL.

2 Related Works

2.1 Offline Reinforcement Learning

Offline RL tackles the “distribution-shift dilemma”: maximize return while staying inside the support of a fixed dataset. (Levine et al. 2020) Previous works have implemented this high-level objective in diverse ways through behavioral regularization (Nair et al. 2020; Fujimoto and Gu 2021; Tarasov et al. 2023), conservatism (Kumar et al. 2020), in-sample maximization (Kostrikov, Nair, and Levine 2021; Xu et al. 2023; Garg et al. 2023), out-of-distribution detection (Yu et al. 2020; Kidambi et al. 2020; An et al. 2021; Nikulin et al. 2023), dual RL (Lee et al. 2021; Sikchi et al. 2023), and generative modeling (Chen et al. 2021; Janner, Li, and Levine 2021; Janner et al. 2022; Park, Li, and Levine 2025). While these methods show promise, they treat trajectories independently, overlooking optimal behavior reconstruction from suboptimal segments. ASTRO addresses this via dynamics-guided trajectory stitching, ensuring both novelty and feasibility.

2.2 Trajectory Stitching

Recently, many works have explored the trajectory stitching problem given offline data in both implicit and explicit ways. Some methods execute stitchability implicitly during planning, using a Q-function (Kim et al. 2024), condition flow (Luo et al. 2025), dynamics model dreaming (Zhou et al. 2023), or Temporal-Distance based graph (Baek et al. 2025). Another category of solution is based on data augmentation. Recent advances in generative models like Diffusion (Song et al. 2020; Ho, Jain, and Abbeel 2020) have enabled high-quality augmentation. Some works focus on high-reward transition (Lu et al. 2023), some directly generate trajectory (Lee et al. 2024; Li et al. 2024). To improve generation quality and diversity, suitable for learning policy, (Yang and Wang 2025; Qing et al. 2025) explore different generation directions, (Lee and Choi 2025) use temporal distance latent space, (Jackson et al. 2024) performing guidance to narrow policy shift. There are also model-based variants, using dynamics model to perform reachable constrained roll-out (Yu et al. 2020; Kidambi et al. 2020; Yu et al. 2021; Char et al. 2022; Park and Lee 2024; Hepburn and Montana 2022). Unlike these approaches, ASTRO uniquely combines temporal-distance-based target selection with explicit dynamics modeling and rollout deviation feedback to generate novel yet dynamics-consistent trajectories.

3 Preliminaries

3.1 Offline Reinforcement Learning

We consider an infinite-horizon Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, \rho_0, p, r, \gamma \rangle$, where \mathcal{S} and \mathcal{A} denote the state and action spaces, ρ_0 is the initial state distribution, p the transition dynamics, r the reward function, and $\gamma \in (0, 1)$ the discount factor. At each timestep t , the agent selects an action $a_t \sim \pi(\cdot | s_t)$, receives reward r_t , and transitions to the next state $s_{t+1} \sim p(\cdot | s_t, a_t)$, thus forming a trajectory $\tau = (s_0, a_0, r_0, s_1, \dots)$. The objective is to learn a policy $\pi^* = \arg \max_{\pi} \mathbb{E}_{\tau \sim \pi} [\sum_{t=0}^{\infty} \gamma^t r_t]$.

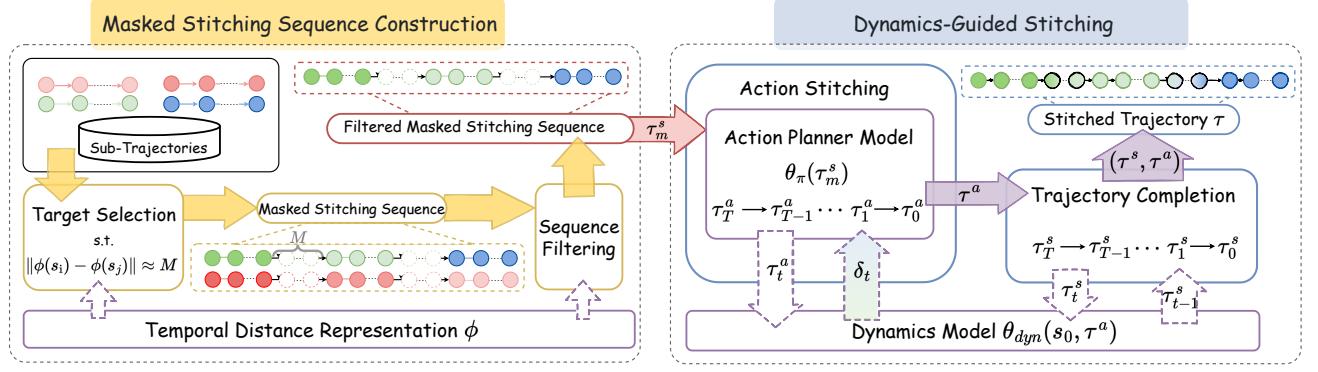


Figure 2: Overview of the **ASTRO** pipeline: (a) stitch target selection based on Temporal Distance Representation (TDR) for temporal coherence. (b) dynamics-guided trajectory completion using diffusion models with rollout deviation for ensuring feasible, high-quality trajectories.

In *offline RL* (Levine et al. 2020), the agent is given a fixed dataset $\mathcal{D} = \{\tau_i\}_{i=1}^N$ collected by an unknown behavior policy π_β and cannot interact with the environment further. A learning algorithm is applied to \mathcal{D} and returns a policy π_θ . Its performance is evaluated by:

$$J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]. \quad (1)$$

Offline RL algorithms must balance return maximization with staying close to the data distribution to avoid distributional shift and value overestimation (Kostrikov, Nair, and Levine 2021; Kidambi et al. 2020; Kumar et al. 2020).

3.2 Temporal Distance Representation

When datasets consist of fragmented trajectories, identifying proper *source–target* pairs becomes critical for trajectory stitching. We adopt the *Temporal Distance Representation* (TDR) proposed by (Park, Kreiman, and Levine 2024; Bae, Park, and Lee 2024; Sun, Qian, and Miao 2024; Lee and Kwon 2025), which embeds each state s into a latent space H via a mapping $\psi : \mathcal{S} \rightarrow H$ such that

$$d^*(s, g) = \|\psi(s) - \psi(g)\|_2 \quad (2)$$

approximates the minimum number of environment steps required to reach g from s .

Learning ψ can be formulated as a goal-conditioned value function estimation problem (Park, Kreiman, and Levine 2024), optimized with an expectile TD loss over offline triples (s, s', g) :

$$\mathcal{L}_{\text{TDR}} = \mathbb{E}_{(s, s', g) \sim \mathcal{D}} \left[l_\tau^2(-\mathbf{1}\{s \neq g\} + \gamma V(s', g) - V(s, g)) \right], \quad (3)$$

where $V(s, g) = -\|\psi(s) - \psi(g)\|_2$ and $l_\tau^2(\cdot)$ denotes the expectile regression loss.

By learning temporal consistency across fragmented trajectories, TDR provides a robust distance metric that generalizes beyond behavior policy limitations.

3.3 Diffusion Models

Gaussian diffusion A T -step diffusion model corrupts a clean sample \mathbf{x}_0 into $\mathbf{x}_1, \dots, \mathbf{x}_T$ through $q(\mathbf{x}_t | \mathbf{x}_{t-1}) =$

$\mathcal{N}(\sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I})$. The denoiser ε_θ is trained *either* to predict the added noise with the standard objective $\mathbb{E}_{t, \mathbf{x}_0, \varepsilon} [\|\varepsilon - \varepsilon_\theta(\mathbf{x}_t, t)\|_2^2]$ or *alternatively* to predict the clean sample, minimising $\mathbb{E}_{t, \mathbf{x}_0} [\|\mathbf{x}_0 - \tilde{\mathbf{x}}_t\|_2^2]$, where

$$\tilde{\mathbf{x}}_t = \tilde{\mathbf{x}}_\theta(\mathbf{x}_t, t) = \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}} \quad (4)$$

Following the denoising diffusion framework to predict the clean sample, we first estimate the clean sample $\tilde{\mathbf{x}}_t$ from the noisy input \mathbf{x}_t using the model-predicted noise $\varepsilon_\theta(\mathbf{x}_t, t)$ and then obtain the denoised sample at the previous step $t-1$ by injecting a small amount of noise back to $\tilde{\mathbf{x}}_t$, following the reverse process of the forward diffusion:

$$\hat{\mathbf{x}}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \tilde{\mathbf{x}}_t + \sqrt{1 - \bar{\alpha}_{t-1}} \cdot \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(0, \mathbf{I}). \quad (5)$$

This formulation reuses the predicted clean sample to deterministically guide the generation of $\hat{\mathbf{x}}_{t-1}$, and is widely adopted in DDPM implementations for sampling.

Sampling starts from \mathbf{x}_T and iteratively denoises to \mathbf{x}_0 .

4 Method

We aim to augment offline RL datasets via trajectory stitching, which connects fragmented sub-trajectories into longer, coherent trajectories. Given a source sub-trajectory $\tau_{\text{sub}} = (s_i, s_{i+1}, \dots, s_{i+l-1})$, our goal is to generate a target sub-trajectory τ'_{sub} by inserting a masked segment MASK that bridges the two. This masked segment is completed by a generative model, which predicts the missing transitions to ensure dynamic consistency.

However, existing stitching approaches often rely on behavior-cloned rollouts and Euclidean distance metrics to select stitching targets. These heuristics tend to produce temporally incoherent connections, violate environment dynamics, and remain constrained to the behavior policy distribution, thereby limiting their effectiveness.

To overcome these limitations, our method ASTRO selects stitching targets using a learned Temporal Distance Representation (TDR), where latent-space distances approximate temporal distance in environment steps. Given a source sub-trajectory, ASTRO identifies target states approximately M steps away in TDR space, inserts masked

transitions, and filters candidate masked Stitching sequences based on TDR-step consistency. A diffusion-based planner then completes the masked segment by generating action sequences guided by a long-horizon dynamics model, ensuring feasible and temporally consistent rollouts.

4.1 ASTRO Stitch Pipeline

Masked stitching Sequence Construction Effective trajectory stitching critically depends on the selection of fragments that are both novel and consistent with environment dynamics. We perform target selection in temporal space using a TDR encoder $\psi : \mathcal{S} \rightarrow \mathbb{R}^d$, which maps states into a latent space where Euclidean distances approximate optimal temporal differences. This process involves two main components: target selection and sequence filtering.

Target Selection To bridge sub-trajectories, we identify a target state s_{target} approximately M steps away in TDR space from the terminal state s_{end} of the current sub-trajectory:

$$s_{\text{target}} = \arg \min_{s \in \mathcal{D}} \|\psi(s_{\text{end}}) - \psi(s)\|_2 - M \quad (6)$$

We then insert a mask sequence MASK of length M between the source and target sub-trajectories. The resulting masked stitching sequence is structured as:

$$\tau_m^s = (\tau_{\text{sub}}^0, \text{MASK}, \tau_{\text{sub}}^1, \text{MASK}, \dots) \quad (7)$$

By leveraging TDR’s distance approximation $|\psi(s_i) - \psi(s_j)|_2 \approx d^*(s_i, s_j)$ and its ability to encode generalizable temporal reachability, this approach identifies coherent and reachable targets beyond the support of the behavior policy, enabling dynamic-consistent stitching.

Sequence Filtering To further ensure smooth and reliable stitching, we apply TDR-based distance filtering to prune unsuitable stitching sequences. For each candidate stitching sequence, we sample k random state pairs (s_m, s_n) both within and across sub-trajectories, then compute the expected temporal distance bias:

$$\mathbb{E}[\Delta_d] = \mathbb{E}[|(m-n) - \|\psi(s_m) - \psi(s_n)\|_2|] \quad (8)$$

Sequences where $\mathbb{E}[\Delta_d] > \Delta_{\text{thresh}}$ are discarded. This enforces local temporal consistency and prevents stitching over structurally inconsistent subsequences (filtering algorithm details in Appendix A).

Dynamics-Guided Stitching

Action stitching Given a masked stitching sequence τ_m^s , ASTRO’s stitch planner θ_π generates an action trajectory τ^a via an adaptive denoising process:

$$\hat{\tau}_{t-1}^a \sim p_{\theta_\pi} \left(\hat{\tau}_{t-1}^a \mid \hat{\tau}_t^a, \tau_m^s, t, \delta(\tilde{\tau}_0^{a,(t+1)}) \right). \quad (9)$$

To ensure dynamic consistency, we introduce Rollout Deviation Feedback δ . Given a noisy state sequence τ_t^s and predicted action sequence τ^a , we compute the deviation between the desired target states and the predicted rollout generated via a learned diffusion dynamics model θ_{dyn} , which denoises state sequence following:

$$\hat{\tau}_{t-1}^s \sim p_{\theta_{\text{dyn}}} \left(\hat{\tau}_{t-1}^s \mid \hat{\tau}_t^s, \tau_{\text{aug}}^a, s_0, t \right) \quad (10)$$

$$\delta(\tau^a) = \left\| \tau^s - \tilde{\tau}_{\theta_{\text{dyn}}}^s(\tau_t^s, s_0, \tau^a, t) \right\|_2^2, \quad t \sim \mathcal{U}(0, T), \quad (11)$$

Here, τ^s refers either to the masked stitching sequence τ_m^s or a full trajectory τ , depending on context. This trajectory-level feedback enables iterative refinement of τ^a , guiding the denoising process toward feasible and reachable actions under the environment’s dynamics.

State Sequence Generation Next, we use the dynamics model θ_{dyn} to roll out the predicted action sequence and generate a dynamics-consistent state sequence τ_{aug}^s . The denoising formulation is:

$$\hat{\tau}_{t-1}^s \sim p_{\theta_{\text{dyn}}} \left(\hat{\tau}_{t-1}^s \mid \hat{\tau}_t^s, \tau_{\text{aug}}^a, s_0, t \right) \quad (12)$$

By iteratively denoising, the model reconstructs the complete state trajectory τ_{aug}^s , ensuring temporal coherence aligned with dynamics.

Trajectory Completion The final augmented trajectory $\tau_{\text{aug}} = (\tau_{\text{aug}}^s, \tau_{\text{aug}}^a)$ is added to the augmentation buffer \mathcal{D}_{aug} . During training, we progressively update the dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\text{aug}}$. The augmented dataset is then used to train the policy using standard offline RL algorithms (implementation details in Appendix A).

4.2 Model Implementation and Training

We now present the architecture and training methodology for the dynamics model θ_{dyn} and the stitch planner θ_π . The planner learns to generate dynamics-consistent action sequences by denoising noisy inputs with guidance from the learned dynamics model.

Dynamics Diffusion Model To provide reliable dynamics feedback and rollouts, we first train a sequence-level dynamics diffusion model θ_{dyn} to reconstruct full state trajectories, conditioned on the initial state s_0 and the corresponding action sequence τ^a . The model is trained to minimize the diffusion reconstruction loss:

$$\begin{cases} \mathcal{L}_{\text{diff}}(\theta_{\text{dyn}}) = \mathbb{E}_{t, \tau_s, \tau_a} \left[\left\| \tau^s - \tilde{\tau}_{\theta_{\text{dyn}}}^s(\tau_t^s, s_0, \tau^a, t) \right\|_2^2 \right] \\ \tilde{\tau}_{\theta_{\text{dyn}}}^s = \frac{1}{\sqrt{\alpha_t}} (\tau_t^s - \sqrt{1 - \bar{\alpha}_t} \epsilon_{\theta_{\text{dyn}}}(\tau_t^s, s_0, \tau^a, t)) \end{cases} \quad (13)$$

where τ_t^s denotes the state trajectory corrupted by forward noise at diffusion step t , s_0 is the initial state, τ^a represents the associated action sequence, and t indicates the diffusion step.

This sequence-level formulation explicitly models long-term dynamics, enabling effective state rollouts without accumulated compounding errors.

Stitch Planner Training The stitch planner θ_π learns to denoise a noisy action sequence and generate a goal-directed action plan aligned with the learned dynamics model. Training is supervised via trajectory reconstruction and rollout deviation minimization.

Adaptive Reconstruction Loss We leverage the Rollout Deviation Feedback τ_m^s from Sec. 4.1) to guide denoising. The adaptive self-correction loss is formulated as:

$$\begin{aligned}\mathcal{L}_{\text{sc}}(\theta_\pi) = & \mathbb{E}_{t, \tau_0} \left[\|\tau^a - \tilde{\tau}_{\theta_\pi}^a(\tau_t^a, \tau_m^s, t, \text{sg}[\delta(\tau^a)])\|_2^2 \right. \\ & \left. + \|\tau^a - \tilde{\tau}_{\theta_\pi}^a(\text{sg}[\hat{\tau}_{t-1}^a], \tau_m^s, t-1, \text{sg}[\delta(\tilde{\tau}_t^a)])\|_2^2 \right]\end{aligned}\quad (14)$$

where $\hat{\tau}_{t-1}^a$ and $\tilde{\tau}_t^a$ follows Eq.5, $\text{sg}[\cdot]$ denotes the stop-gradient operator. The first term encourages accurate prediction despite model error, while the second facilitates recursive correction between denoising steps. This loss encourages robust denoising by incorporating trajectory-level deviation signals, helping the planner iteratively correct toward feasible action sequences.

Deviation Regularization To further promote feasible and reliable stitching, we penalize generated actions whose predicted rollouts exhibit larger deviation from the target states than those produced by the ground-truth actions:

$$\mathcal{L}_{\text{reg}}(\theta_\pi) = \mathbb{E}_{t, \tau_0} \left[(\delta(\tilde{\tau}_0^{a,(t)}) - \delta(\tau^a))_+ \right], \quad (15)$$

where $(x)_+ = \max(0, x)$ is the ReLU operator. This term penalizes the stitch planner only when generated plans degrade reachability relative to expert actions. This regularization, applied within the model’s confidence region, improves stitching reliability and enhances downstream task performance.

Joint Training Objective The final objective for training the stitch planner is a weighted combination of the self-correction and regularization losses:

$$\mathcal{L}(\theta_\pi) = \mathcal{L}_{\text{sc}}(\theta_\pi) + \alpha \mathcal{L}_{\text{reg}}(\theta_\pi), \quad (16)$$

where α is a hyperparameter that controls the trade-off between trajectory accuracy and dynamics-aligned feasibility.

5 Experiments

5.1 Experimental Setup

Benchmarks We primarily evaluate ASTRO on OG-Bench (Park et al. 2024), a challenging benchmark consisting of robotic locomotion and manipulation tasks characterized by sparse goal-achievement rewards. We select the reward-based, single-task variants that are compatible with standard offline RL algorithms, specifically including three manipulation and five locomotion tasks.

For locomotion tasks, we utilize the `stitch` variant datasets, which consist of short, fragmented trajectory segments that require effective stitching to learn coherent long-horizon behavior.

Additionally, we evaluate on six widely used AntMaze tasks from the D4RL benchmark (Fu et al. 2020), allowing broader comparison against standard baselines. The environment details are illustrated in Appendix B.

Baselines We compare ASTRO against two state-of-the-art trajectory augmentation methods: (1) DiffStitch (Li et al. 2024): A trajectory stitching approach that generates new rollouts via start–goal conditioned diffusion models. (2) SynthER (Lu et al. 2023): A reward-guided diffusion method that augments the replay buffer by synthesizing high-reward transitions. All methods are evaluated under two popular offline RL algorithms: (1) IQL (Kostrikov, Nair, and Levine 2021): A conservative one-step algorithm based on advantage-weighted behavior cloning. (2) FQL (Park, Li, and Levine 2025): A more expressive algorithm that uses flow-matching-based action sampling. To ensure fair comparisons, we use identical training protocols, network architectures, and diffusion backbones across all methods (see Appendix B for details).

Evaluation Protocol Agents are trained for a fixed number of gradient steps, and we report performance using the final checkpoint (rather than selecting the best), to avoid early-stopping bias.

Each reported result represents the mean standard deviation over eight random seeds per task. Complete Evaluation details are provided in Appendix B.

5.2 Results and Analysis

We systematically evaluate ASTRO across four key research questions that assess its (1) overall performance in offline RL, (2) underlying improvement mechanisms, (3) reliability of temporal-distance-based target selection, and (4) quality of trajectories generated via dynamics-guided completion. Our results demonstrate how ASTRO’s components synergistically improve trajectory augmentation in offline RL.

Q1: How significantly does ASTRO enhance offline RL performance? Table 1 shows that ASTRO consistently achieves substantial performance gains across locomotion and manipulation tasks with dense or sparse reward, significantly outperforming existing methods. On average, ASTRO boosts scores by **+26.2%** under IQL ($36.08 \rightarrow 45.52$) and **+18.4%** under FQL ($55.52 \rightarrow 65.71$).

ASTRO excels across distinct performance regimes:

- **Moderate-performing tasks** (baseline scores between 20 and 80 on the original dataset): ASTRO achieves substantial improvements, increasing average scores by **+15.83** (IQL) and **+18.57** (FQL), significantly outperforming DiffStitch (**+5.73** IQL, **+5.10** FQL) and SynthER (**+2.30** IQL, **+2.07** FQL).
- **Low-return scenarios** (baseline scores below 20 on the original dataset): ASTRO demonstrates strong performance recovery, improving IQL from 3.05 to 13.60 and FQL from 11.40 to 20.20. In comparison, DiffStitch yields marginal gains (IQL: $3.05 \rightarrow 3.05$, FQL: $11.40 \rightarrow 12.95$), while SynthER leads to performance degradation (IQL: $3.05 \rightarrow 2.60$, FQL: $11.40 \rightarrow 10.05$).
- **High-performing tasks** (baseline scores above 80 on the original dataset): Even in strong-performing regimes, ASTRO achieves further gains of **+7.70** (IQL) and **+3.80** (FQL), surpassing DiffStitch (**-2.50** IQL, **-0.65** FQL) and SynthER (**+0.90** IQL, **-1.50** FQL).

Table 1: Comparison of ASTRO against baselines (DiffStitch, SynthER) across OGBench and D4RL benchmarks, evaluated with offline RL algorithms IQL and FQL. Results highlight ASTRO’s average task performance improvements in various locomotion and manipulation tasks.

Task	IQL				FQL			
	Original	ASTRO	DiffStitch	SynthER	Original	ASTRO	DiffStitch	SynthER
OGBench Maze Stitch	ant-large-v0	26.2	51.7	35.0	31.1	29.2	57.3	33.1
	ant-giant-v0	0	0	0	0	5.3	10.4	3.3
	humanoid-medium-v0	29.7	31.4	28.3	31.2	17.5	30.0	22.6
	humanoid-large-v0	2.4	12.6	2.2	1.4	2.7	3.5	2.9
	antsoccer-arena-v0	3.7	14.6	3.9	3.8	22.4	49.3	28.5
	maze-avg	12.40	22.06	13.88	13.50	15.42	30.10	18.08
OGBench Manipulation Play	scene-v0	31.7	40.6	32.1	27.5	94.3	97.0	91.4
	cube-single-v0	81.5	89.2	79.0	82.4	88.0	92.9	89.6
	cube-double-v0	2.4	2.5	2.6	0.5	36.5	45.4	40.1
	manipulation-avg	38.53	44.10	37.9	36.8	72.93	78.43	73.7
D4RL avg (6 ant-mazes)		57.3	70.4	65.3	63.5	78.2	88.6	85.0
Total avg		36.08	45.52	39.03	37.93	55.52	65.71	58.93
								56.01

Table 2: **Ablation study** on ASTRO’s stitching mechanism using FQL across four challenging OGBench locomotion tasks. Column labels denote, *Ori*: original dataset performance, *ASTRO*: our full method, *Rand*: random target selection, *Euc*: Behavioral Pre-generation + Euclidean-distance-based target selection, *MB*: model-based rollout without guidance, *SI*: state planner with inverse dynamics. ASTRO consistently outperforms all ablations, demonstrating the importance of both Temporal-distance-space target selection and dynamics-guided trajectory stitching.

Task	Ori	ASTRO	w/ Rand	w/ Pre+Euc	w/ MB	w/ SI
AntMaze-Large-v0	29.2	57.3	36.2	41.3	46.5	35.9
AntMaze-Giant-v0	5.3	10.4	7.2	4.4	6.3	8.1
HumanoidMaze-Medium-v0	17.5	30.0	23.9	21.7	28.3	11.5
AntSoccer-Arena-v0	22.4	49.3	30.9	36.0	46.1	33.3
Average	18.60	36.75	24.55	25.85	31.80	22.20

Importantly, ASTRO consistently maintains or improves performance across all environments, unlike DiffStitch and SynthER, which occasionally degrade results. This highlights ASTRO’s robustness and reliability in trajectory augmentation.

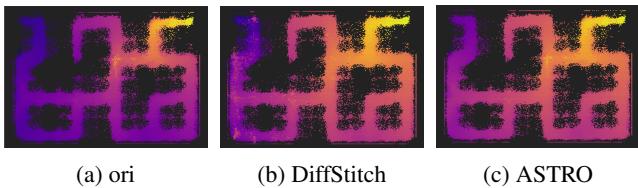


Figure 3: Value-function heatmaps on *antmaze-large*. Warmer colors indicate higher Q-values; the goal is located in the upper-right corner. ASTRO facilitates effective reward propagation, yielding substantial improvements in Q_{mean} : +16.59 (from -111.17 to -94.58) for IQL and +7.06 (from -87.55 to -80.49) for FQL. In contrast, DiffStitch yields only marginal gains of +3.82 and +0.43, respectively.

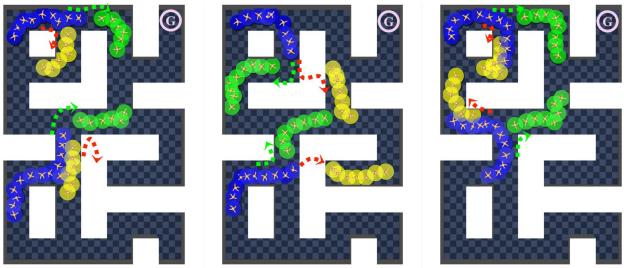
Q2: Why does ASTRO work? We use the *antmaze-large* environment from OGBench as a

case study to investigate the mechanisms driving ASTRO’s improvements.

In sparse-reward environments with complex dynamics, high-value signals are often concentrated near goal regions, making it difficult for value functions to propagate effectively. ASTRO mitigates this by injecting dynamics-consistent rollouts that expand the set of reachable, high-reward states, thereby promoting more effective Q-value propagation. As shown in Figure 3, ASTRO significantly improves the distribution of Q-values across the state space, resulting in higher average Q-values and enhanced downstream policy performance. Notably, ASTRO achieves a +9.71 increase in Q_{mean} compared to baseline methods.

Q3: How critical is Temporal distance Space for stitch target selection? We analyze the importance of Temporal Distance (TD) space in target selection as follows using *antmaze-large* as a representative case.

Performance Impact As illustrated in Table 2 Replacing TD-space with alternative selection strategies leads to significant degradation. Using Euclidean distance, FQL drops from 36.75 → 25.85 (-10.9). Uniform random selection



(a) Stitch Smoothness (b) Stitch Feasibility (c) Stitch Novelty

Figure 4: Analysis of Temporal-Distance-space selection in `antmaze-large`: (a) Temporal-Distance-space selection maintains low angular deviation for smooth paths; (b) Temporal-Distance-space selection Avoids infeasible target for dynamics violations; (c) Explores beyond behavior distribution boundaries

Table 3: Quantitative analysis of geometric properties for selected stitch source-target pairs in `antmaze-large` environment, comparing TDR target selection against baseline methods, highlighting improvements in angular deviation and trajectory curvature.

Selection Method	$ \Delta\theta \downarrow$	Curvature \uparrow
Pre-Gen+Eucli	3.147 ± 2.621	0.697 ± 0.325
original traj	2.089 ± 1.455	0.764 ± 0.153
TD-Space	1.253 ± 0.451	0.934 ± 0.079

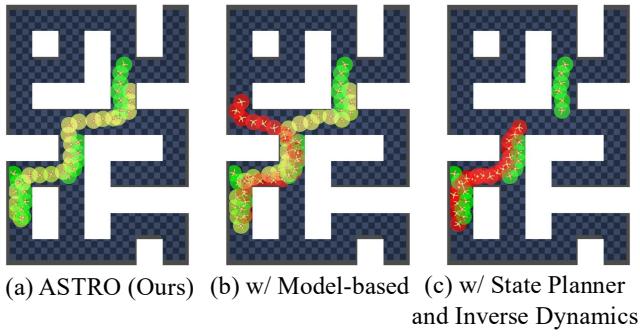
performs even worse, reducing FQL to 24.55 (-12.2).

Robust Target selection TD space captures temporal coherence, improving stitching quality. In contrast, Euclidean matching often yields trajectories with: (1) high angular deviation ($|\Delta\theta|$, average direction change between segments) and (2) low curvature (inverse turning radius, indicating sharper turns). These result in non-smooth paths that violate dynamics constraints (Table 3, Fig.4(a); calculation details in Appendix). It may also select unreachable goals, such as those behind walls (Fig.4(b))(detailed metric calculation in Appendix C).

Distributional Generalization TD-based selection enables ASTRO to go beyond the behavior policy, choosing targets outside the support of the original dataset (Fig. 4). This overcomes the limitations of pre-generated, behavior-cloned rollouts and facilitates more diverse and effective stitching.

Table 4: Evaluation of trajectory completion quality on the `antmaze-large` task, comparing full ASTRO’s dynamics-guided completion against state-based planners (SI), sequence-model methods (MB), and some ASTRO variants, assessed by action and state prediction errors as well as dynamics violation frequency.

Method	τ^a MSE	τ^s MSE	τ_m^s MSE	Dyn_Violation
SI	0.226	0.954	0.695	17.4
MB	0.141	0.782	0.452	12.3
ASTRO (w/ L_{reg})	0.138	0.724	0.391	9.2
ASTRO (w/ L_{sc})	0.129	0.723	0.402	8.5
ASTRO	0.103	0.657	0.351	5.3



(a) ASTRO (Ours) (b) w/ Model-based (c) w/ State Planner and Inverse Dynamics

Figure 5: Completion from different generation pipelines.

Q4: Does dynamics-guided Stitching enhance trajectory quality?

Feasible Completions Table 4 shows that the sequence model alone reduces action MSE by 37.6% and dynamics violations by 29.3% compared to inverse dynamics-based stitching. ASTRO further improves on this, reducing action MSE to 0.103 and violations to 5.3% via training regularization and inference-time self-conditioning.

High Target Reachability We assess target reachability by the mean squared error (MSE) between reached states and target states. As shown in Table 4, ASTRO consistently achieves lower deviation (0.35 MSE) compared to both model-based (0.45 MSE) and inverse dynamics-based stitching (0.70 MSE), which suffer from greater misalignment due to inaccurate action prediction. This confirms that ASTRO not only ensures feasibility but also enhances target reachability through its dynamics-guided completion(detailed metric calculation in Appendix C).

Visual Evidence Visual evidence in Figure 5 demonstrates ASTRO’s dynamics-guided stitching advantages: (a) ASTRO successfully completes trajectories by leveraging its diffusion-based planner with rollout deviation feedback, ensuring dynamic consistency; (b) The sequence model-based method fails to reach targets due to lack of dynamics-aware refinement; (c) The inverse dynamics planner produces state-action misalignment and infeasible plans without explicit dynamics modeling. These results validate how ASTRO’s dynamics-guided completion enables feasible, high-reachability stitching.

6 Conclusion

We presented ASTRO, an adaptive trajectory stitching framework for offline reinforcement learning that addresses key limitations of existing augmentation methods. By leveraging a Temporal Distance Representation for temporally coherent target selection, and employing a dynamics-guided diffusion planner with rollout deviation feedback, ASTRO generates trajectories that are both novel and feasible. This enables more effective value propagation across fragmented sub-trajectories, improving long-horizon policy learning.

References

- Agarwal, R.; Schuurmans, D.; and Norouzi, M. 2020. An optimistic perspective on offline reinforcement learning. In *International conference on machine learning*, 104–114. PMLR.
- An, G.; Moon, S.; Kim, J.-H.; and Song, H. O. 2021. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34: 7436–7447.
- Bae, J.; Park, K.; and Lee, Y. 2024. Tldr: Unsupervised goal-conditioned rl via temporal distance-aware representations. *arXiv preprint arXiv:2407.08464*.
- Baek, S.; taegeon park; Park, J.; Oh, S.; and Kim, Y. 2025. Graph-Assisted Stitching for Offline Hierarchical Reinforcement Learning. In *Forty-second International Conference on Machine Learning*.
- Char, I.; Mehta, V.; Villaflor, A.; Dolan, J. M.; and Schneider, J. 2022. Bats: Best action trajectory stitching. *arXiv preprint arXiv:2204.12026*.
- Chen, L.; Lu, K.; Rajeswaran, A.; Lee, K.; Grover, A.; Laskin, M.; Abbeel, P.; Srinivas, A.; and Mordatch, I. 2021. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34: 15084–15097.
- Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *arXiv preprint arXiv:2004.07219*.
- Fujimoto, S.; and Gu, S. S. 2021. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34: 20132–20145.
- Garg, D.; Hejna, J.; Geist, M.; and Ermon, S. 2023. Extreme q-learning: Maxent rl without entropy. *arXiv preprint arXiv:2301.02328*.
- Hepburn, C. A.; and Montana, G. 2022. Model-based trajectory stitching for improved offline reinforcement learning. *arXiv preprint arXiv:2211.11603*.
- Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.
- Jackson, M. T.; Matthews, M. T.; Lu, C.; Ellis, B.; Whiteson, S.; and Foerster, J. 2024. Policy-guided diffusion. *arXiv preprint arXiv:2404.06356*.
- Janner, M.; Du, Y.; Tenenbaum, J. B.; and Levine, S. 2022. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*.
- Janner, M.; Li, Q.; and Levine, S. 2021. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34: 1273–1286.
- Kidambi, R.; Rajeswaran, A.; Netrapalli, P.; and Joachims, T. 2020. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823.
- Kim, S.; Choi, Y.; Matsunaga, D. E.; and Kim, K.-E. 2024. Stitching sub-trajectories with conditional diffusion model for goal-conditioned offline rl. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 13160–13167.
- Kostrikov, I.; Nair, A.; and Levine, S. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*.
- Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33: 1179–1191.
- Lee, D.; and Kwon, M. 2025. Temporal Distance-aware Transition Augmentation for Offline Model-based Reinforcement Learning. *arXiv preprint arXiv:2505.13144*.
- Lee, J.; Jeon, W.; Lee, B.; Pineau, J.; and Kim, K.-E. 2021. Optidice: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, 6120–6130. PMLR.
- Lee, J.; Yun, S.; Yun, T.; and Park, J. 2024. Gta: Generative trajectory augmentation with guidance for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37: 56766–56801.
- Lee, K.; and Choi, J. 2025. State-Covering Trajectory Stitching for Diffusion Planners. *arXiv preprint arXiv:2506.00895*.
- Levine, S.; Kumar, A.; Tucker, G.; and Fu, J. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.
- Li, G.; Shan, Y.; Zhu, Z.; Long, T.; and Zhang, W. 2024. Diffstitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. *arXiv preprint arXiv:2402.02439*.
- Lu, C.; Ball, P.; Teh, Y. W.; and Parker-Holder, J. 2023. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36: 46323–46344.
- Luo, Y.; Mishra, U. A.; Du, Y.; and Xu, D. 2025. Generative trajectory stitching through diffusion composition. *arXiv preprint arXiv:2503.05153*.
- Nair, A.; Gupta, A.; Dalal, M.; and Levine, S. 2020. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*.
- Nikulin, A.; Kurenkov, V.; Tarasov, D.; and Kolesnikov, S. 2023. Anti-exploration by random network distillation. In *International conference on machine learning*, 26228–26244. PMLR.
- Park, K.; and Lee, Y. 2024. Model-based Offline Reinforcement Learning with Lower Expectile Q-Learning. *arXiv preprint arXiv:2407.00699*.
- Park, S.; Frans, K.; Eysenbach, B.; and Levine, S. 2024. Ogbench: Benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*.
- Park, S.; Kreiman, T.; and Levine, S. 2024. Foundation policies with hilbert representations. *arXiv preprint arXiv:2402.15567*.
- Park, S.; Li, Q.; and Levine, S. 2025. Flow q-learning. *arXiv preprint arXiv:2502.02538*.

- Qing, Y.; Chen, S.; Chi, Y.; Liu, S.; Lin, S.; and Zou, C. 2025. BiTrajDiff: Bidirectional Trajectory Generation with Diffusion Models for Offline Reinforcement Learning. *arXiv preprint arXiv:2506.05762*.
- Sikchi, H.; Zheng, Q.; Zhang, A.; and Niekum, S. 2023. Dual rl: Unification and new methods for reinforcement and imitation learning. *arXiv preprint arXiv:2302.08560*.
- Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.
- Sun, C.; Qian, H.; and Miao, C. 2024. Cudc: A curiosity-driven unsupervised data collection method with adaptive temporal distances for offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 15145–15153.
- Tarasov, D.; Kurenkov, V.; Nikulin, A.; and Kolesnikov, S. 2023. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 11592–11620.
- Xu, H.; Jiang, L.; Li, J.; Yang, Z.; Wang, Z.; Chan, V. W. K.; and Zhan, X. 2023. Offline rl with no ood actions: In-sample learning via implicit value regularization. *arXiv preprint arXiv:2303.15810*.
- Yang, Q.; and Wang, Y. 2025. Rtdiff: Reverse trajectory synthesis via diffusion for offline reinforcement learning. In *International Conference on Learning Representations (ICLR)*.
- Yu, T.; Kumar, A.; Rafailov, R.; Rajeswaran, A.; Levine, S.; and Finn, C. 2021. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34: 28954–28967.
- Yu, T.; Thomas, G.; Yu, L.; Ermon, S.; Zou, J. Y.; Levine, S.; Finn, C.; and Ma, T. 2020. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33: 14129–14142.
- Zhou, Z.; Zhu, C.; Zhou, R.; Cui, Q.; Gupta, A.; and Du, S. S. 2023. Free from bellman completeness: Trajectory stitching via model-based return-conditioned supervised learning. *arXiv preprint arXiv:2310.19308*.

Appendix A: Detailed Algorithms

Algorithm 1: Dynamics-Guided Stitch Planner Training

- 1: Assume pretrained dynamics model θ_{dyn} is fixed
- 2: Initialize stitch planner θ_π
- 3: **repeat**
- 4: Sample clean trajectory (τ^s, τ^a) from dataset
- 5: Sample diffusion timestep $t \sim \mathcal{U}(\{1, \dots, T\})$
- 6: Corrupt actions to obtain noisy τ_t^a via forward diffusion
- 7: Compute rollout deviation using eq.11
- 8: Compute self-conditioned loss $\mathcal{L}_{\text{self-cond}}$ using eq.15
- 9: Compute regularization loss \mathcal{L}_{reg} using eq.14
- 10: Compute total loss $\mathcal{L}(\theta_\pi) = \mathcal{L}_{\text{self-cond}} + \mathcal{L}_{\text{reg}}$; update θ_π
- 11: **until** converged
- 12: **Return** trained stitch planner θ_π

Algorithm 2: Dynamics-Guided Stitch Planner Inference

- 1: Given masked stitching sequence τ_m^s
- 2: Initialize noisy action sequence $\hat{\tau}_T^a \sim \mathcal{N}(0, I)$
- 3: rollout deviation $\delta(\tilde{\tau}_0^{a,(T+1)}) = 0$
- 4: **for** $t = T$ down to 1 **do**
- 5: Compute rollout deviation $\delta(\tilde{\tau}_0^{a,(t+1)})$ using eq.11
- 6: Sample $\hat{\tau}_{t-1}^a$ using eq.9
- 7: **end for**
- 8: **Return** refined action sequence $\hat{\tau}_0^a$

Algorithm 3: Policy Training with Adaptive Weighted Trajectory Stitch Augmentation

- 1: Given pretrained dynamics model θ_{dyn} , stitch planner θ_π , and offline dataset \mathcal{D}
- 2: Initialize policy π_θ , critic Q_ϕ , and stitched buffer $\mathcal{B}_{\text{stitch}} = \emptyset$
- 3: **for** each offline RL training iteration **do**
- 4: Perform temporal-space stitch target selection on \mathcal{D} to generate source-target state pairs
- 5: Use stitch planner θ_π to generate stitched actions $\hat{\tau}_0^a$
- 6: Use dynamics model θ_{dyn} to roll out stitched states $\hat{\tau}_0^s$ given s_0 and $\hat{\tau}_0^a$
- 7: Form stitched trajectory $\hat{\tau}_i = (\hat{\tau}_0^s, \hat{\tau}_0^a)$
- 8: Sample mini-batch from \mathcal{D} and buffer $\mathcal{B}_{\text{stitch}}$
- 9: Update policy π_θ and critic Q_ϕ using offline RL algorithm
- 10: **end for**
- 11: **Return** optimized policy π_θ

Appendix B: Environments and Model Implementation

B.1 Environments and Datasets

We use the OGBench and D4RL datasets for evaluation. For OGBench maze, we use stitch dataset to validate our

Algorithm 4: TDR-based Sequence Filter

Require: Stitching sequence τ_m^s ; TDR encoder ψ ; sample count k ; threshold Δ_{thresh}

Ensure: $\text{keep} \in \{\text{true}, \text{false}\}$

- 1: $\mathcal{B} \leftarrow \emptyset$ ▷ store distance biases
- 2: **for** $t = 1$ to k **do**
- 3: Randomly sample state pairs (s_m, s_n) from τ_m^s
- 4: $d_{\text{step}} \leftarrow |m - n|$ ▷ temporal distance
- 5: $d_{\text{TDR}} \leftarrow \|\psi(s_m) - \psi(s_n)\|_2$
- 6: $\mathcal{B} \leftarrow \mathcal{B} \cup \{|d_{\text{step}} - d_{\text{TDR}}|\}$
- 7: **end for**
- 8: $\mathbb{E}[\Delta_d] \leftarrow \frac{1}{k} \sum_{b \in \mathcal{B}} b$ **return** $\text{keep} \leftarrow (\mathbb{E}[\Delta_d] \leq \Delta_{\text{thresh}})$

method. Figure shows the difference between stitch dataset and navigate and explore dataset, which consists of short trajectories that RL must stitch between trajectories to enable effective planning, which causing the low performance of the offline RL baselines without augmentation. The performance improvement in all environments shows our augmentation is effective.

Environments

Antmaze. In this dataset, an agent controls a quadruped Ant robot with 8 degrees of freedom (DoF). The objective is to navigate through various maze configurations to reach a designated goal location, requiring simultaneous mastery of high-level pathfinding and low-level locomotion control. The environments are provided in three distinct maze sizes: medium, large, and giant, with larger mazes specifically crafted to evaluate extensive long-horizon planning capabilities. Observations are represented by a 29-dimensional state vector, capturing the robot's two-dimensional position (x-y coordinates) and detailed joint-related information. Simulations are executed within the MuJoCo physics simulator.

Humanoidmaze. This dataset introduces increased complexity by utilizing a Humanoid robot with 17 degrees of freedom (DoF). The task involves navigating medium and large mazes, significantly testing long-horizon planning and sophisticated locomotion control due to the robot's more intricate dynamics. The state observation consists of a 376-dimensional vector, capturing comprehensive joint positions, velocities, and various proprioceptive signals. All simulations are performed using the MuJoCo physics engine.

Antsoccer-area. This environment integrates manipulation and locomotion, tasking an Ant robot with 8 DoF to simultaneously dribble a ball and navigate within an arena setting. This task evaluates the agent's capacity for coupled object manipulation and precise locomotion. Observations are state-based and include a 40-dimensional vector covering the agent's positional information, joint states, and interactions with the ball. The environment is simulated in MuJoCo.

Scene-play. The Scene-play dataset examines sequential decision-making skills requiring complex reasoning about multiple interactive objects, including a drawer, window, button locks, and a cube. Data is collected through open-loop, non-Markovian scripted interactions, mimicking play-

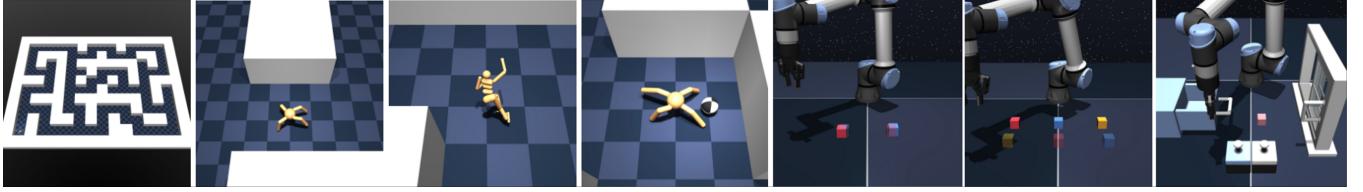


Figure 6: OGbench environments

like behavior with diverse everyday objects. Observations are state-based, capturing comprehensive object poses and robotic joint states. Simulated in MuJoCo.

Cube-single-play. This dataset assesses an agent’s capability to manipulate a single cube using flexible pick-and-place strategies. Data collection involves randomized open-loop scripted interactions, emphasizing robust generalization across various manipulation tasks. Observations include detailed positions and orientations of the cube alongside robotic joint states. The simulations are conducted using MuJoCo.

Cube-double-play. Building upon single-object manipulation, this dataset introduces complexity by requiring the coordinated handling and stacking of two cubes. Data is collected via open-loop scripted interactions, emphasizing sequential reasoning and compositional task generalization. Observations consist of positions, orientations of both cubes, and detailed robotic joint states. The environment simulations utilize MuJoCo.

Dataset Types

Stitching. The stitching dataset type specifically evaluates the agent’s proficiency in assembling partial trajectories into cohesive, optimal plans. Data comprises short goal-reaching trajectories generated by noisy expert policies, navigating to randomly sampled intermediate goals.

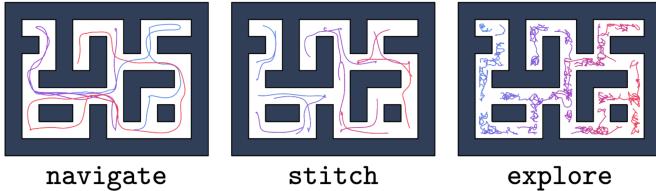


Figure 7: Difference between stitch dataset and navigate and explore dataset.

Manipulation. The manipulation dataset type evaluates sequential decision-making capabilities, emphasizing reasoning about past actions to inform future plans. The play-style dataset employs open-loop, non-Markovian scripted interactions to simulate diverse object manipulations. The partial dataset derives from human demonstrations originally captured using a VR teleoperation interface.

Goal Specification for Evaluation Goal specification and evaluation follow the established OGbench protocol, utilizing default tasks and assessing agent performance across 8 random seeds per environment.

Evaluation Metric Performance evaluation utilizes the standard success rate metric.

B.2 Model Implementation

B.2.1 Base RL Implementation We implement two state-of-the-art offline RL algorithms as base learners for ASTRO evaluation:

IQL (Implicit Q-Learning): A conservative one-step algorithm that performs advantage-weighted behavior cloning. IQL learns a value function $V(s)$ and advantage function $A(s, a)$ through implicit Q-learning, then extracts the policy via:

$$\pi(a|s) \propto \exp(\beta A(s, a)) \quad (17)$$

where β is a temperature parameter controlling conservatism. IQL avoids distributional shift by staying close to the behavior policy while maximizing expected returns.

FQL (Flow Q-Learning): A more expressive algorithm that uses flow-matching-based action sampling. FQL learns a flow model that directly parameterizes the policy distribution, enabling more flexible action generation compared to IQL’s conservative approach. The flow model is trained to match the optimal action distribution through continuous normalizing flows, providing better exploration of the action space while maintaining policy improvement guarantees.

Both algorithms are implemented with identical network architectures (3-layer MLPs with 256 hidden units) and training protocols (learning rate 3×10^{-4} , batch size 256, 1M gradient steps) to ensure fair comparison across augmentation methods.

We follow the hyperparameter settings from FQL for all environments and algorithms, shown in table 5.

All experiments use identical random seeds and evaluation protocols across methods for fair comparison.

B.2.2 Diffusion Implementation

Network Architecture. For both stitch planner and dynamics model, we use a diffusion transformer architecture based on DiT for sequence modeling in high-dimensional continuous control. Specifically, we adopt the publicly released DiT model architecture (from facebookresearch) as the denoising network within our diffusion framework. The network leverages a transformer backbone designed for conditional generation, allowing it to scale to long horizons and complex dynamics. To allow different field of state and action, we use unified concat of state and action sequence to form as trajectory sequence.

Horizon. We adopt task-specific fixed horizons to align with the complexity and planning requirements of each environment. The specific horizon lengths used during training and inference are summarized in Table 6.

Table 5: IQL and FQL hyperparameters used in our experiments.

Task	IQL α	FQL α	Discount γ
ant-large-v0	10	10	0.99
ant-giant-v0	10	10	0.995
humanoid-medium-v0	10	30	0.995
humanoid-large-v0	10	30	0.995
antsoccer-arena-v0	1	10	0.995
scene-v0	10	100	0.99
cube-single-v0	1	300	0.99
cube-double-v0	0.3	100	0.99
D4RL avg (6 ant-mazes)	–	10 (umaze/medium), 3 (large)	0.99

Table 6: Horizon lengths used for each environment.

Benchmark	Environment	Horizon
OGBench AntMaze Stitch	antmaze-large-v0	180
	antmaze-giant-v0	180
OGBench HumanoidMaze Stitch	humanoid-medium-v0	340
	humanoid-large-v0	340
OGBench AntSoccer Stitch	antsoccer-arena-v0	180
	antsoccer-arena-v0	180
OGBench scene	–	180
	single	180
OGBench cube	double	180
	–	180
D4RL AntMaze Stitch		

B.2.3 TDR Selection Implementation We adopt a goal relabeling strategy similar to prior works, with a key modification: we explicitly exclude the trivial case where the current state and goal are identical ($s = g$), since our temporal distance formulation satisfies $V(s, s) = 0$ by construction.

At each training step, the goal state g is sampled using the following mixture strategy. With probability 0.625, g is sampled from a future state along the same trajectory, where the offset follows a geometric distribution over time indices. With probability 0.375, g is uniformly sampled from the full replay buffer, enabling global goal matching. This sampling scheme ensures that $s = g$ is never selected, and reflects the original hyperparameter design that balances local temporal reasoning with global generalization.

The temporal distance predictor $V(s, g)$ is modeled as a feed-forward multilayer perceptron (MLP) with architecture (512, 512, 512), outputting a 32-dimensional latent embedding or 64-dimensional latent embedding (for humanoid-maze). To stabilize training and improve representation consistency, we apply Layer Normalization to the MLP hidden activations.

For selecting the masked stitching sequence, we choose a unified stitching chain of 5 to different tasks, and use $l/M = 1$ to facilitate balance mask and sub-trajectory information. Ablation against hyperparameters is shown in Appendix D.

B.2.4 Baselines Implementation For fair comparison, we use the same implementation of our DiT to evaluate Diff-stitch and SynthER. We tune the low-to-high reward strategy to noised sampling to ensure balanced augmentation for DiffStitch in sparse reward environments. For other hyper-

parameters such as Euclidean threshold, we follow the original implementation.

Appendix C: Metrics for Geometric Consistency

To quantify the geometric quality of stitched trajectories, we employ three metrics that capture local smoothness, directional coherence and dynamics consistency:

Directional Change ($\Delta\theta$). We compute the change in heading angle between consecutive segments to measure abrupt directional shifts. Given three consecutive positions s_{t-1}, s_t, s_{t+1} , the angle difference is computed as:

$$\Delta\theta_t = \angle(s_t - s_{t-1}, s_{t+1} - s_t), \quad (18)$$

where $\angle(\cdot, \cdot)$ computes the signed angle between two 2D vectors.

Trajectory Curvature. We further measure the local curvature to assess how sharply the agent turns along the path. The curvature κ_t at timestep t is defined by:

$$\kappa_t = \frac{\|s_{t+1} - 2s_t + s_{t-1}\|}{\|s_{t+1} - s_t\|^2 + \|s_t - s_{t-1}\|^2}, \quad (19)$$

where $\|\cdot\|$ denotes the Euclidean norm. This metric penalizes high-frequency oscillations and is sensitive to trajectory smoothness.

Dynamics Violation Rate. To evaluate the physical feasibility of stitched transitions, we introduce the *dynamics violation rate*, which quantifies how often a transition (s, a, s') violates the underlying environment dynamics.

Table 7: Ablation results on antmaze-large-v0. Policy performance and selection latency (in seconds) are reported.

(a) Filter Threshold			(b) Chain Length			(c) Horizon L		(d) Mask M/l	
Threshold	Performance	Time	Length	Success (%)	Time	L	$\text{MSE} \downarrow$	M/l	$\text{MSE} \downarrow$
5	48.9	2.1	3	52.1	2.4	120	0.673	1/3	0.672
3	57.3	4.5	5	57.3	6.5	160	0.657	1/1	0.657
2	58.2	12.4	7	59.6	25.6	200	0.698	3/1	0.715

Given a stitched pair (s, s') (without known action), we sample n random actions $\{a_i\}_{i=1}^n$ from the environment’s action space and evaluate whether any of them can plausibly generate s' from s . Specifically, we define the violation criterion as:

$$\mathcal{V}(s, s') = \begin{cases} 1, & \text{if } \forall a_i \in \mathcal{A}, \|f(s, a_i) - s'\| > \delta, \\ 0, & \text{otherwise,} \end{cases} \quad (20)$$

where $f(s, a_i)$ is the next state predicted by stepping the environment forward with action a_i , and δ is a small threshold for matching the next state.

The dynamics violation rate is computed as the average across all stitched transitions:

$$\text{Violation Rate} = \frac{1}{N} \sum_{j=1}^N \mathcal{V}(s_j, s'_j), \quad (21)$$

where N is the number of stitched transitions being evaluated.

Appendix D: Ablation Study

In this section, we conduct comprehensive ablation studies on critical design choices that directly impact ASTRO’s ability to generate distributionally novel and dynamics-consistent trajectories. All experiments are conducted on antmaze-large-v0, a challenging long-horizon navigation task with fragmented demonstration data that exemplifies the core problems addressed by ASTRO and FQL as RL base agent.

D.1 Ablation on TDR-based Stitch Target Selection

We first analyze how TDR design choices affect the identification of distinct and reachable stitch targets, which is fundamental to ASTRO’s ability to create novel trajectory connections. The time below is the average selection time of 10 samples.

Filter Threshold. The TDR filter threshold determines the quality of selected stitch targets by controlling how strict the temporal-distance constraints are. Tighter thresholds (lower values) ensure that only highly compatible state pairs are considered for stitching, leading to more dynamics-consistent augmented trajectories and improved policy performance. However, this comes at the cost of significantly increased computational overhead due to more stringent filtering requirements. A threshold of 3 provides the optimal trade-off between trajectory quality and computational efficiency, as shown in Table 7(a).

Chain Length. Longer subtrajectory Chain Length enable more sophisticated trajectory compositions and can potentially discover more diverse stitching paths through the state space. While longer Chain Length provide marginal performance improvements by enabling more complex trajectory augmentations, they suffer from exponential increases in selection latency due to the combinatorial growth of possible chain configurations. A Chain Length of 5 achieves the best balance between augmentation diversity and practical computational constraints, as demonstrated in Table 7(b).

D.2 Ablation on Dynamics-Guided Diffusion Planning

Next, we analyze hyperparameters that affect the quality of dynamics-guided action sequence generation, which is crucial for ensuring the feasibility and reachability of stitched trajectories. We use generated rollout state MSE as metric.

Diffusion Horizon L . The trajectory horizon length directly impacts the model’s ability to capture long-range dependencies and generate coherent action sequences for trajectory stitching. Moderate horizons ($L = 160$) provide sufficient context for effective dynamics modeling without overfitting to specific trajectory patterns. Excessively long horizons can lead to degraded performance due to increased model complexity and potential overfitting to suboptimal trajectory segments, as evidenced in Table 7(c).

Subtrajectory Masking Ratio M/l . This ratio controls the balance between global trajectory structure preservation and local action sequence generation during diffusion training. A balanced setting ($M/l = 1$) ensures that the model learns to generate action sequences that are both locally coherent and globally consistent with the overall trajectory dynamics. Extreme ratios either under-constrain the generation process (leading to dynamics violations) or over-constrain it (strict target reaching goal), as shown in Table 7(d).

Appendix E: Additional visualizations

Here we present some selection examples for visualization, showing TDR-based robust and distinct selection for trajectory stitching.

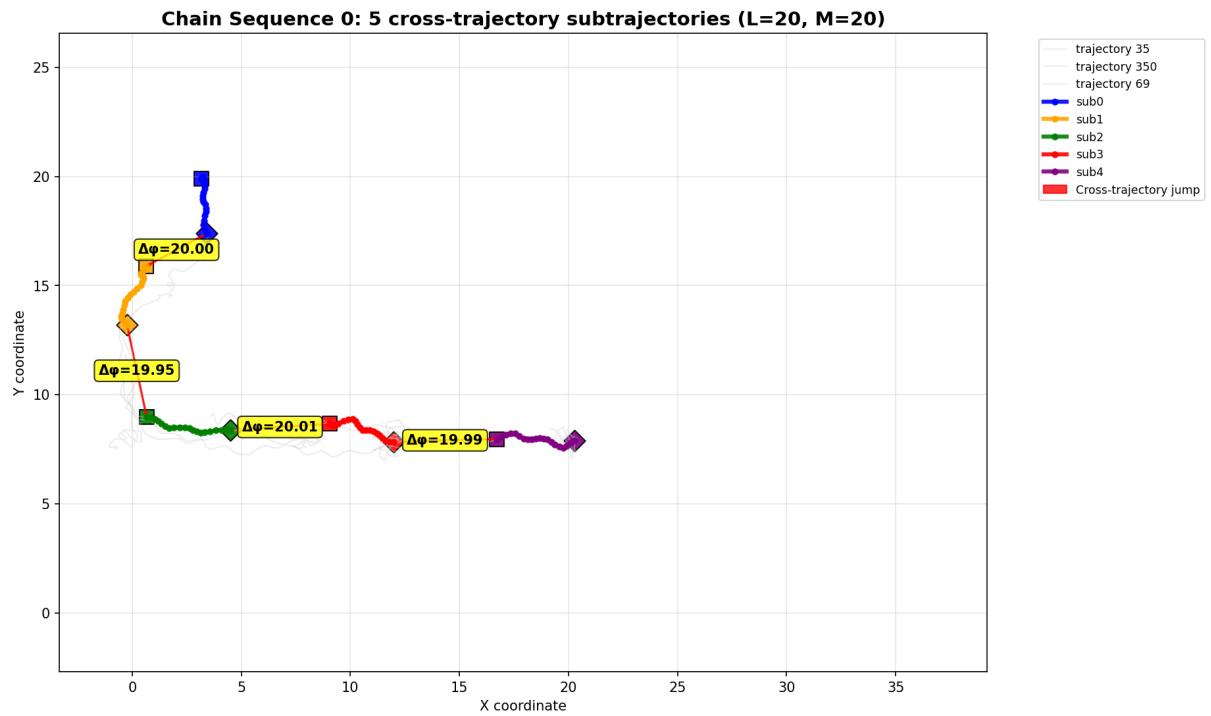


Figure 8: Visualization of selection results 0.

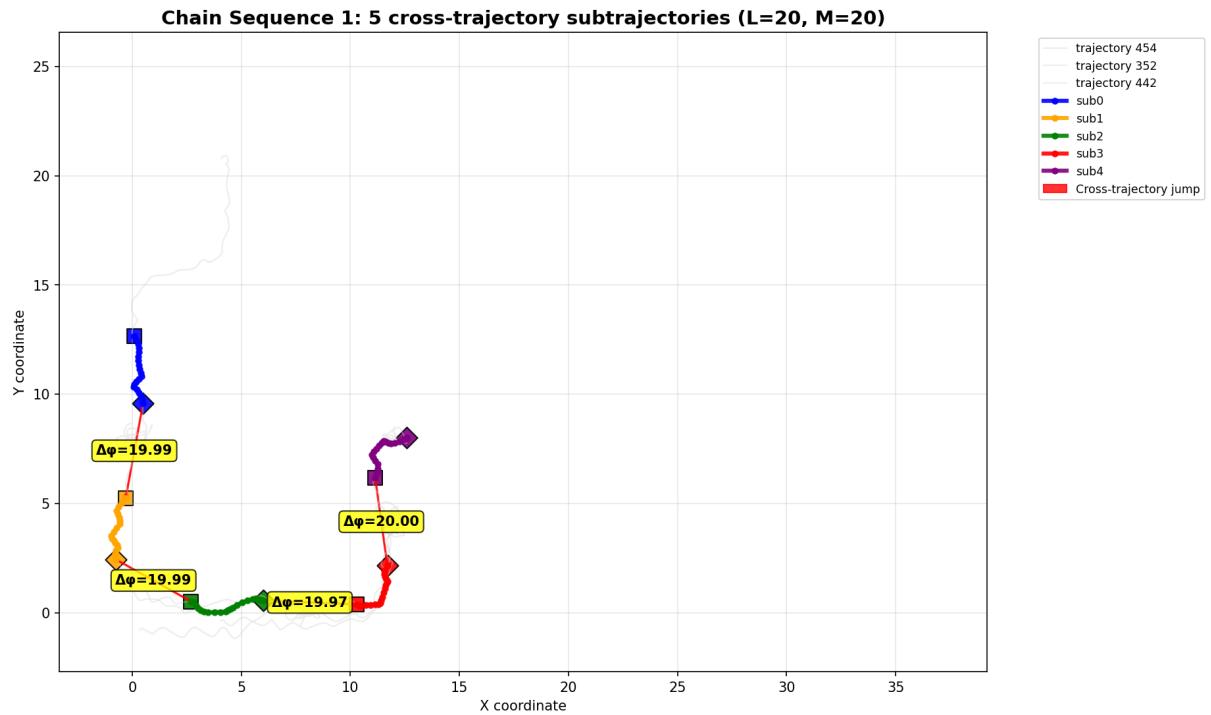


Figure 9: Visualization of selection results 1.

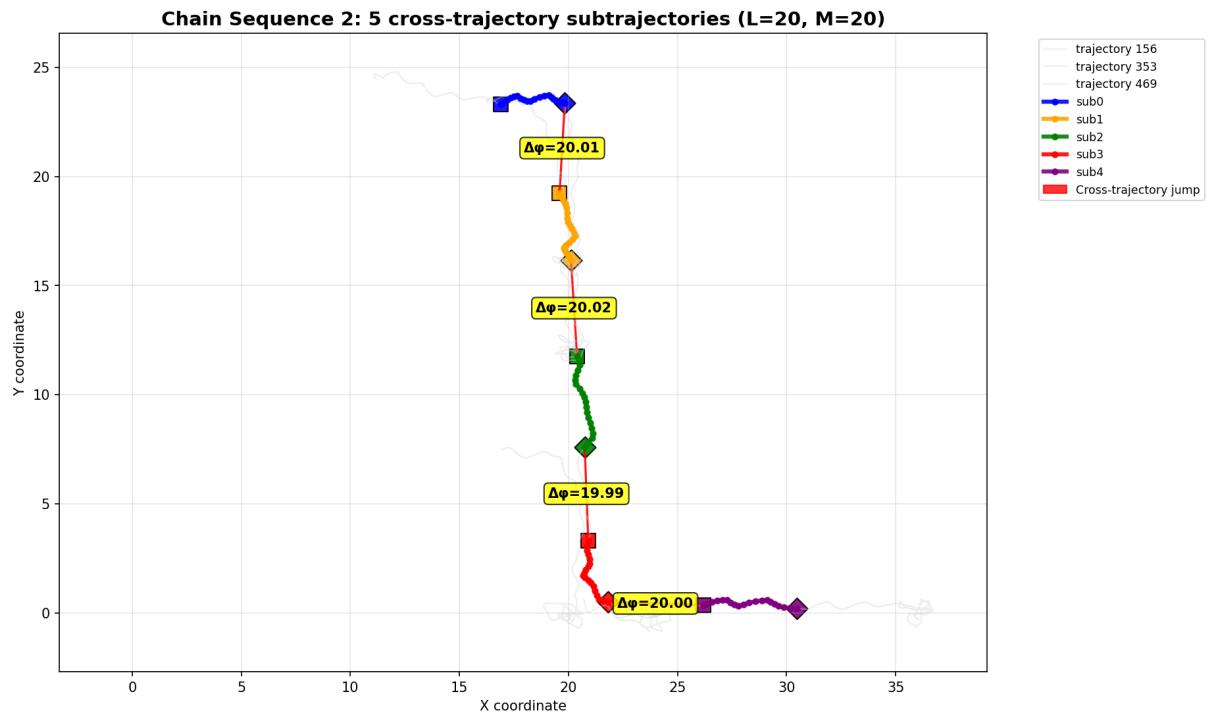


Figure 10: Visualization of selection results 2.

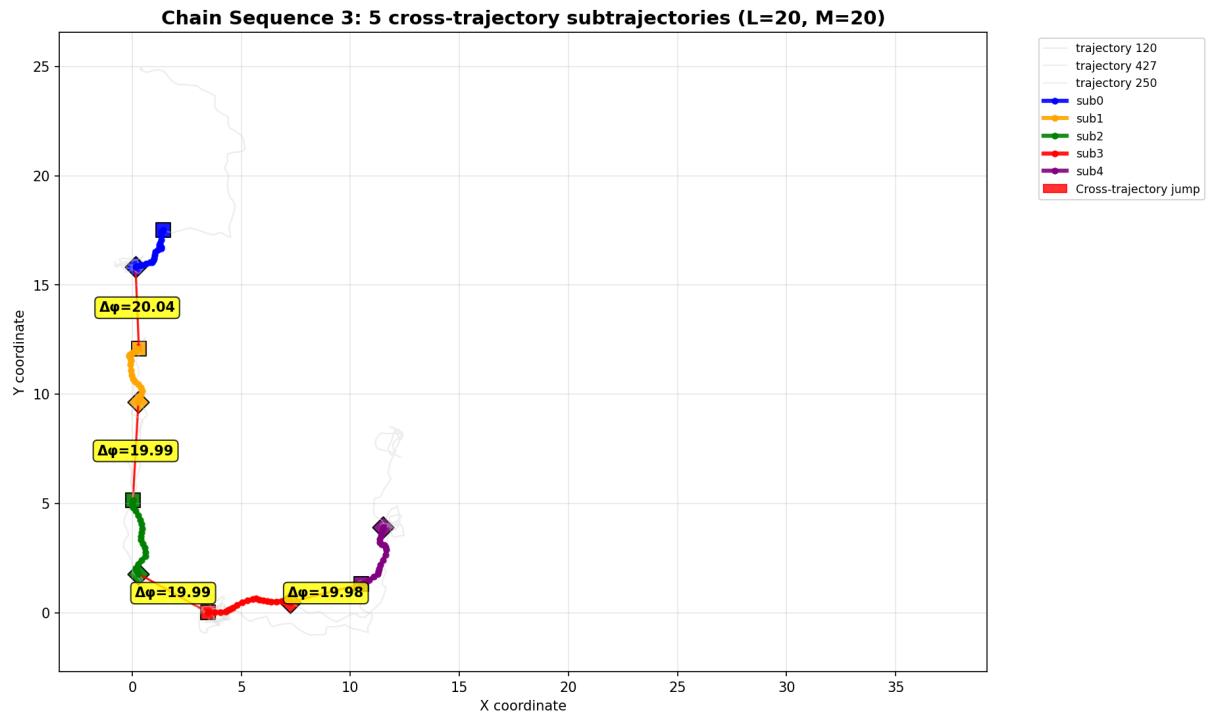


Figure 11: Visualization of selection results 3.

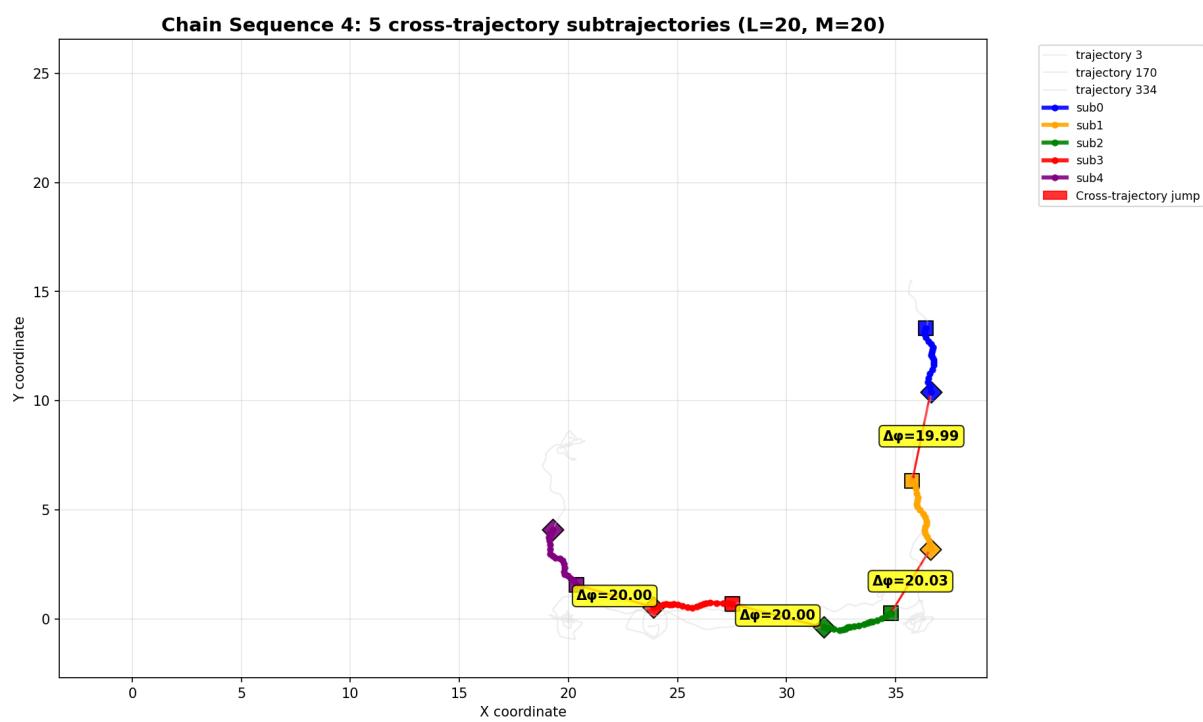


Figure 12: Visualization of selection results 4.