Safe Flow Q-Learning: Offline Safe Reinforcement Learning with Reachability-Based Flow Policies

Mumuksh Tayal*, **Manan Tayal***, **Ravi Prakash**

Keywords: Safe reinforcement learning, offline reinforcement learning, flow matching, Hamilton-Jacobi reachability.

Summary

Safe offline reinforcement learning seeks reward-maximizing control from static datasets under strict safety constraints. We propose **Safe Flow Q-Learning (SafeFQL)**, which extends FQL to the safe setting by combining a Hamilton-Jacobi reachability-inspired safety value function with an efficient one-step flow policy for safe action selection without rejection sampling at deployment. Empirically, SafeFQL trades modestly higher offline training cost for substantially lower inference latency than diffusion-style safe generative baselines, making it attractive for real-time safety-critical control. Across boat navigation and all Safe Velocity based Gymnasium MuJoCo tasks, SafeFQL matches or exceeds prior offline safe RL performance while reducing constraint violations.

Contribution(s)

1. We propose **Safe Flow Q-Learning (SafeFQL)**, a reachability-aware extension of Flow Q-Learning for safe offline reinforcement learning that learns an expressive one-step policy without iterative denoising or rejection sampling at inference.
   Context: The method is evaluated in offline settings with fixed datasets and does not claim online-training safety guarantees.
2. We provide a computation-time analysis showing that SafeFQL trades modestly higher offline training cost for substantially lower inference latency than diffusion-style safe generative baselines, supporting real-time deployment in safety-critical loops.
   Context: Latency gains are reported for the evaluated implementations, hardware, and benchmark settings.
3. Across boat navigation and all Safety Gymnasium MuJoCo tasks, SafeFQL co-optimizes safety and reward, matching or exceeding prior offline safe RL performance while reducing constraint violations.
   Context: Empirical findings are established on the reported benchmarks and may vary across datasets and task distributions.

Safe Flow Q-Learning: Offline Safe Reinforcement Learning with Reachability-Based Flow Policies

Mumuksh Tayal[*1], **Manan Tayal**[*1], **Ravi Prakash**[1]
mumukshtayal@iisc.ac.in, manantayal@iisc.ac.in, ravipr@iisc.ac.in

[1] **Indian Institute of Science, India**
[*]**Denotes equal contribution**

Abstract

Offline safe reinforcement learning (RL) seeks reward-maximizing policies from static datasets under strict safety constraints. Existing methods often rely on soft expected-cost objectives or iterative generative inference, which can be insufficient for safety-critical real-time control. We propose **Safe Flow Q-Learning (SafeFQL)**, which extends FQL to safe offline RL by combining a Hamilton–Jacobi reachability-inspired safety value function with an efficient one-step flow policy. SafeFQL learns the safety value via a self-consistency Bellman recursion, trains a flow policy by behavioral cloning, and distills it into a one-step actor for reward-maximizing safe action selection without rejection sampling at deployment. Empirically, SafeFQL trades modestly higher offline training cost for substantially lower inference latency than diffusion-style safe generative baselines, which is advantageous for real-time safety-critical deployment. Across boat navigation, and Safety Gymnasium MuJoCo tasks, SafeFQL matches or exceeds prior offline safe RL performance while substantially reducing constraint violations. Project page for the work is available here.

1 Introduction

Constrained reinforcement learning (CRL) methods incorporate safety objectives during policy learning, but most established approaches rely on extensive online interaction and repeated environment rollouts (Achiam et al., 2017; Altman, 2021; Alshiekh et al., 2018; Zhao et al., 2023). This dependence is problematic in safety-critical domains, where training-time failures are costly and many systems do not have sufficiently faithful simulators to absorb risky exploration. As also reflected in safe-RL benchmarks and datasets (Liu et al., 2024), the online setting can expose both training and deployment to unacceptable safety risk. These limitations motivate a shift toward offline policy synthesis from logged data, including offline RL and imitation-style pipelines (Levine et al., 2020; Kumar et al., 2020). However, even in offline settings, many methods still enforce safety through expected cumulative penalties or Lagrangian dual updates, yielding *soft* constraint satisfaction rather than strict state-wise guarantees (Xu et al., 2022; Ciftci et al., 2024; Stooke et al., 2020). Such formulations can be insufficient when a single violation is unacceptable, and the safety-performance trade-off becomes particularly brittle when safety-critical transitions are sparse in static datasets (Lee et al., 2022).

Control-theoretic safety methods provide a complementary perspective with stronger notions of state-wise safety. Control Barrier Functions (CBFs) (Ames et al., 2014) and Hamilton–Jacobi (HJ) reachability (Bansal et al., 2017; Fisac et al., 2019) can encode forward invariance and worst-case safety explicitly. Yet, classical grid-based HJ methods face the curse of dimensionality (Mitchell, 2005). In addition, many practical CBF/HJ-inspired learning pipelines require either known dynamics

or a learned dynamics surrogate to compute safety derivatives and synthesize actions (e.g., through QP-based filtering(Ames et al., 2017)). When dynamics are unknown, model-learning errors can propagate into safety estimates and policy decisions, particularly under dataset shift and out-of-support actions, which weakens practical robustness in purely offline settings (Tayal et al., 2025a;b). Recent offline safety frameworks also report this trade-off explicitly: learned models can enable scalable controller synthesis, but they may become a dominant error source for high-confidence safety if not carefully calibrated (Tayal et al., 2025b). In parallel, safe generative-policy methods have emerged to improve action expressivity under offline distributional constraints. Sequence-model approaches such as the Constrained Decision Transformer (CDT) condition generation on return and cost budgets (Liu et al., 2023), while diffusion-based methods model multimodal action distributions and can better represent complex behavior support in static datasets (Janner et al., 2022).These advances are important because safety-critical datasets are often heterogeneous and multimodal, where unimodal Gaussian actors can fail to recover rare but important safe maneuvers. However, current safe generative policies still face practical bottlenecks: sequence-model conditioning is indirect for step-wise safety control, and diffusion-style policies require iterative denoising and often additional rejection sampling to reliably pick safe high-value actions at test time, increasing latency and deployment complexity (Liu et al., 2023; Zheng et al., 2024).

At the same time, recent progress in offline RL suggests that improving value learning alone is often insufficient: even with a reasonably accurate critic, extracting an effective policy remains non-trivial (Park et al., 2024). Flow matching provides a useful alternative to diffusion-style generation by learning a continuous transport (velocity-field) map from noise to actions, enabling expressive policy classes with simpler sampling dynamics (Lipman et al., 2023). Building on this idea, Flow Q-Learning (FQL) in unconstrained offline RL separates flow-based behavior modeling from one-step RL policy optimization, so the final actor can be optimized efficiently without backpropagating through iterative generation (Park et al., 2025). Extending this idea to safety-critical offline RL is not a trivial drop-in adaptation. In the safe setting, policy extraction must simultaneously (i) maximize reward, (ii) remain inside a safety-feasible region under future evolution, and (iii) avoid excessive conservatism that degrades performance.

Motivated by this, we propose **Safe Flow Q-Learning (SafeFQL)**, an offline safe RL framework that combines reachability-inspired safety value learning with one-step flow policy extraction. SafeFQL learns a safety value function that captures feasibility through a Bellman-style recursion over offline data, and trains a distilled one-step actor that is directly optimized by Q-learning while regularized toward the behavior-supported flow policy. This avoids recursive backpropagation through iterative generative sampling and removes the need for rejection sampling at deployment, while retaining expressive action modeling.

To summarize, our main contributions are:

- We formulate **SafeFQL**, a reachability-aware extension of FQL for safe offline RL that learns an expressive one-step policy without iterative denoising or rejection sampling at inference.
- We provide a dedicated **computation-time analysis** showing that, while SafeFQL may incur higher offline training cost, it delivers substantially lower inference latency than diffusion-style safe generative baselines, enabling real-time deployment in safety-critical control loops.
- We show that SafeFQL **co-optimizes safety and performance** across custom navigation and Safety Gymnasium benchmarks, consistently achieving lower safety violations while maintaining strong reward relative to prior constrained offline RL and safe generative baselines.

2 Background and Problem Setup

We study safe offline reinforcement learning in environments with hard state constraints. The environment is modelled as a Constrained Markov Decision Process (CMDP), defined by the tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, P, r, \ell, \gamma)$, where $\mathcal{X}$ and $\mathcal{A}$ denote the state and action spaces, $P(x'|x, a)$ denotes the transition probability function defining the system dynamics, $r : \mathcal{X} \to \mathbb{R}$ is the reward function,

$\ell : \mathcal{X} \to \mathbb{R}$ is an instantaneous state-based safety function, typically defined as the negative of signed distance function to failure set $\mathcal{F}$, and $\gamma \in (0, 1)$ is the discount factor. We define the *failure set* $\mathcal{F} := \{x \in \mathcal{X} \mid \ell(x) > 0\}$, which represents unsafe states that must be avoided at all times (e.g., collisions or constraint violations). A trajectory is considered safe if it never enters $\mathcal{F}$. We assume access to an offline dataset $\mathcal{D} = \{(x_t, a_t, r_t, \ell_t, x_{t+1})\}$, collected by an unknown behavior policy, with no further interaction with the environment permitted. Any policy $\pi(a \mid x)$ which induces trajectories $\Gamma = (x_0, a_0, x_1, \dots)$, does it through the transition probability function P.

Given an initial state x, the objective is to compute the maximum achievable discounted return subject to state safety at all future time steps. This requirement can be formalized through the following formulation:

$$\sup_{\pi} \Big[\sum_{k=0}^{\infty} \gamma^k r(x_k) \,\Big|\, x_0 = x \Big] \quad \text{s.t.} \;\; x_t \notin \mathcal{F}, \; \forall t \geq 0. \tag{1}$$

Unlike formulations based on expected cumulative penalties, (1) encodes a *hard safety requirement*, i.e., only policies that admit trajectories remaining entirely outside the failure set are considered feasible. This formulation directly captures safety-critical requirements where even a single violation is unacceptable.

2.1 Generative Policies for Offline RL

To overcome the limitations of traditional limitations for policy extraction, recent literature has investigated generative policy representations in offline RL, such as sequence models and diffusion-based policies (Chen et al., 2021; Janner et al., 2022), along with their extensions to safety-constrained environments (Liu et al., 2023; Lin et al., 2023; Zheng et al., 2024; LIU et al., 2025). Although highly effective at capturing data distributions, diffusion models necessitate the simulation of stochastic processes across numerous discrete time steps during inference. This iterative sampling is computationally burdensome, making real-time deployment in high-frequency control loops particularly challenging. Conversely, flow matching (Lipman et al., 2023; Zhang et al., 2025b; Alles et al., 2025) presents a deterministic alternative. By directly learning the vector field of the generative process, flow matching facilitates highly efficient policy sampling through a single ODE integration. A convenient way to view flow-matching policies is as the time-1 pushforward of a state-conditioned, time-dependent velocity field. Let $v_\theta(t, x, z)$ denote the state-conditioned velocity field and define the flow $\psi_\theta(t, x, z)$ by the ODE

$$\frac{d}{dt}\, \psi_\theta(t, x, z) = v_\theta\big(t, x, \psi_\theta(t, x, z)\big), \qquad \psi_\theta(0, x, z) = z. \tag{2}$$

The corresponding flow policy is defined as the ODE terminal map

$$\mu_\theta(x, z) \coloneqq \psi_\theta(1, x, z) = z + \int_0^1 v_\theta\big(t, x, \psi_\theta(t, x, z)\big)\, dt, \tag{3}$$

which is a deterministic mapping in (x, z) but induces a stochastic policy $\pi_\theta(a \mid x)$ via $z \sim \mathcal{N}(0, I)$. We will dive deeper into this aspect of deterministic mapping of (x, z) in the later sections.

2.2 Safe Offline Reinforcement Learning

Safe reinforcement learning has conventionally relied on online Lagrangian-based constrained optimization and trust-region methods (Chow et al., 2017; Tessler et al., 2018; Stooke et al., 2020; Achiam et al., 2017). However, the necessity for online interaction and the use of soft cost penalties in these approaches have catalyzed a shift toward safe offline RL. Several prominent offline RL methods such as CPQ (Xu et al., 2022) and C2IQL (LIU et al., 2025) attempt to ensure safety by penalizing unsafe actions by restricting the expected cumulative costs below a pre-defined cost limit l, i.e.,

$\max_\pi \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty}\gamma^t r(x_t,a_t)]$ s.t. $\mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty}\gamma^t c(x_t)] \le l$; but these techniques often degrade value estimation and generalization (Li et al., 2023).

Some Hamilton–Jacobi (HJ) reachability based safety frameworks connect HJ reachability with offline RL (Zheng et al., 2024) to identify states that can enter the failure set $\mathcal{F} = \{x : \ell(x) \ge 0\}$ within a given time horizon (Bansal et al., 2017; Fisac et al., 2019). They often define the HJ value as the best worst-time safety margin

$$V_\ell^*(x_0) := \inf_\pi \sup_{t\in[0,T]} \ell(x_t) \quad \text{s.t. } a_t \sim \pi(\cdot \mid x_t), \tag{4}$$

$$\text{Or,} \quad V_\ell^*(x_0) := \max\{\ell(x_0), \inf_\pi V_\ell^\pi(x_1)\} \quad \forall\, t \in \{0,1,2,...\} \tag{5}$$

so that $V_\ell^*(x_0)$ measures the smallest maximum value of ℓ attainable along trajectories from x_0. Intuitively, $V_\ell^*(x_0) > 0$ indicates that even the best policy leads the trajectory inside the failure set (i.e., $\ell(x_t) \ge 0$ for some t), while $V_\ell^*(x_0) < 0$ implies there exists an optimally safe policy that keeps the system in the safe region from state x_0 within the horizon. The classical HJ PDE / Hamiltonian formulation and numerical solution methods are used for computation (Bansal et al., 2017). Such frameworks often use Generative Policy based techniques like DDPM (Zheng et al., 2024) and Flow Matching to learn expressive policies in offline RL. However, such frameworks struggle to extract an exact optimal policy and rather tend to learn a policy which only encourages the desired safety and performance with the use of Advantage Weighted Regression (Peters & Schaal, 2007). And even though AWR is a simple and easy-to-implement approach in offline RL, it is often considered as the least effective policy extraction method (Park et al., 2024), and therefore, many a times has to be accompanied by Rejection Sampling to selectively choose an action that best suites the requirements. Perhaps, a more effective technique for policy extraction can be using Deterministic Policy Gradient with Behavior Cloning (Fujimoto & Gu, 2021) where the policy directly maximizes Q-value function. But using DPG with multi-step denoising based generative policy frameworks like Flow Matching requires backward gradient through the entire reverse denoising process, inevitably introducing tremendous computational costs. Meanwhile, other set of frameworks use Barrier Function based approaches (Wang et al., 2023a; Tayal et al., 2025b) to achieve safety. Unfortunately, these frameworks also come with their own set of limitations. Barrier Functions require knowledge of system dynamics which is generally rare to be known for most systems. Although such frameworks choose to learn the approximate dynamics of the system, they can become a significant source of noise, which can be fatal in safety-critical cases.

To overcome these bottlenecks, recent works have focused on distilling the multi-step generative processes into single-step policies (Prasad et al., 2024; Zhang et al., 2025a; Park et al., 2025). These distilled models are designed to match the action outputs of their full-fledged, multi-step counterparts, yielding fast and accurate performance at a fraction of the computational cost for both training and inference.

3 Safe Flow Q-Learning

Building on the CMDP formulation and the offline safe RL objective introduced in Section 2, this section presents SafeFQL in full detail. The design follows the decoupled learning principle of FQL (Park et al., 2025) where value functions and the policy are trained with separate objectives so that policy optimization is never destabilized by errors in critic bootstrapping. We extend this principle to the safety-constrained setting by introducing a second critic system whose semantics are governed by worst-case reachability rather than cumulative discounted return.

The overall procedure decomposes into three phases: *(i)* learning reward and safety critics from $\mathcal{D}$; *(ii)* fitting a behavior flow teacher and distilling it into a one-step actor; and *(iii)* optimizing the actor under a feasibility-gated objective. These three phases are sequentially dependent, the policy cannot be trained before critics converge. Within each phase, all networks are trained in parallel to convergence. We describe each phase in turn.

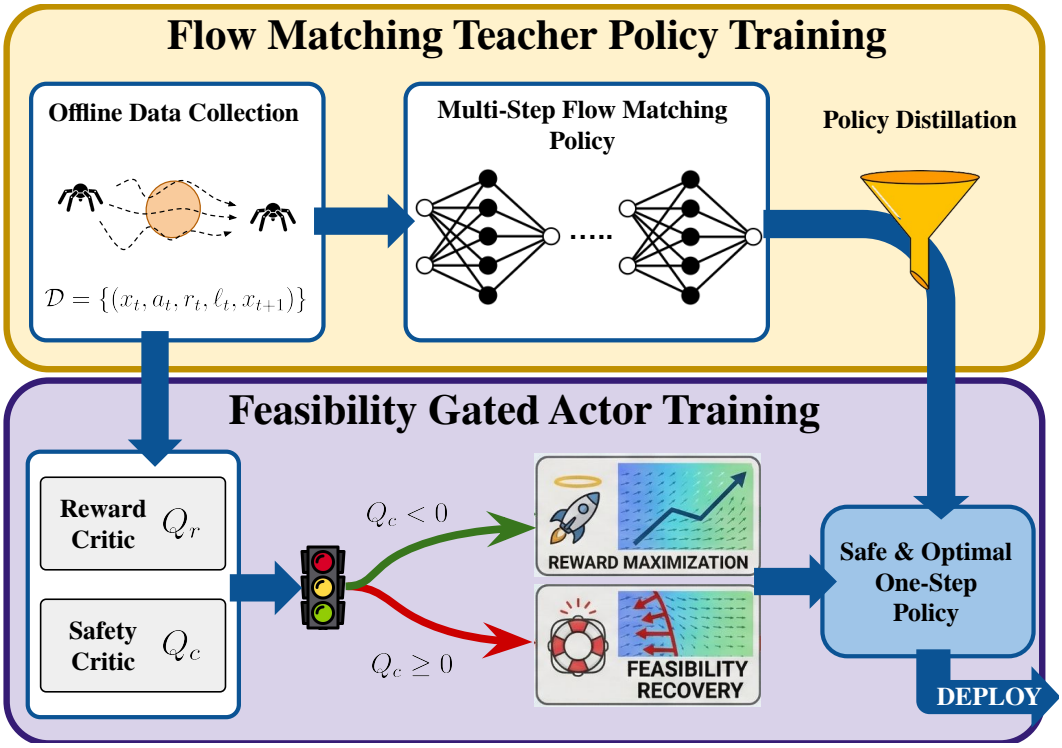

Figure 1: **Framework Overview.** SafeFQL framework proposes a 2 stage safe offline RL approach which uses an efficient one-step flow policy extraction using policy distillation and feasibility gating.

3.1 Learning Reward and Safety Critics

The offline dataset $\mathcal{D} = \{(x_i, a_i, r_i, \ell_i, x'_i)\}_{i=1}^N$ provides tuples of state, action, scalar reward, signed safety signal, and next state. We recall from Section 2 that the safety signal $\ell(x)$ is defined so that $\ell(x) \leq 0$ if and only if $x \notin \mathcal{F}$, i.e., the state is safe. All critic learning is performed entirely within the support of $\mathcal{D}$, so that no out-of-distribution action queries are required.

Reward critics. We train a reward Q-function $Q_r(x, a; \phi_r)$ and a corresponding state-value function $V_r(x; \psi_r)$ using the implicit Q-learning (IQL) approach of Kostrikov et al. (2022). IQL avoids querying the actor during critic updates, which is the primary source of instability in offline actor–critic methods (Fujimoto et al., 2019). The value function V_r approximates the expectile of the Q-value distribution under the behavior policy, and is trained via the asymmetric squared loss

$$\mathcal{L}_{V_r}(\psi_r) = \mathbb{E}_{(x,a)\sim\mathcal{D}}[\mathcal{L}_\tau(Q_r(x, a; \phi_r) - V_r(x; \psi_r))], \tag{6}$$

where $\mathcal{L}_\tau(u) = |\tau - \mathbb{I}(u < 0)|\, u^2$ is the expectile loss with $\tau \in (0.5, 1)$. For τ close to 1 the loss upweights positive residuals, causing V_r to track a high quantile of the in-sample Q-value distribution rather than its mean. This implicitly represents the advantage of actions better than average in the dataset without ever evaluating the policy. Given V_r, the Q-function is updated via one-step Bellman regression against a target network $\bar{V}_r$:

$$y_r = r + \gamma\, \bar{V}_r(x'), \tag{7}$$

$$\mathcal{L}_{Q_r}(\phi_r) = \mathbb{E}_{(x,a,r,x')\sim\mathcal{D}}\left[(Q_r(x, a; \phi_r) - y_r)^2\right]. \tag{8}$$

Target network parameters $\bar{\psi}_r$ are updated via Exponential Moving Average (EMA), details for which are covered in Supplementary Material C.

Safety critics. For the safety constraint, a naive approach would be to train a discounted cumulative cost Q-function $Q_c^{\text{sum}}(x,a) = \mathbb{E}[\sum_t \gamma^t \mathbb{I}\{x_t \in \mathcal{F}\}]$ and penalize its expectation below a threshold, as in standard CMDP Lagrangian methods. This leads to a soft constraint that enforces safety in expectation but cannot prevent individual trajectory violations (Xu et al., 2022; Lee et al., 2022). Moreover, the non-negativity of the cumulative cost makes the threshold a free hyperparameter that must be tuned per task.

SafeFQL instead adopts a reachability-inspired formulation that encodes *worst-case* safety along the trajectory. We define the safety critic $Q_c(x,a)$ as an approximation of the Hamilton–Jacobi feasibility value $V_\ell^*(x_0) = \min_\pi \max_{t\geq 0} \ell(x_t)$ from Section 2, trained via a *max-backup* Bellman recursion (Fisac et al., 2019):

$$y_c(x,a,x') = \max\{\ell(x),\ \gamma\, \bar{V}_c(x')\}\,. \tag{9}$$

The target y_c takes the *maximum* of the immediate safety margin $\ell(x)$ and the discounted future safety value $\gamma\bar{V}_c(x')$. This ensures that a low safety margin at *any* future time step propagates backward to the current state, so that $Q_c(x,a) < 0$ carries a strong meaning: not only is x currently safe, but the predicted future evolution also remains in the safe region under behavior-policy-like actions. Conversely, $Q_c(x,a) \geq 0$ indicates that following the behavior distribution from (x,a) is predicted to eventually enter the failure set $\mathcal{F}$.

The safety Q-function $Q_c(x,a;\phi_c)$ and safety value function $V_c(x;\psi_c)$ are trained with

$$\mathcal{L}_{Q_c}(\phi_c) = \mathbb{E}_{(x,a,\ell,x')\sim\mathcal{D}}\left[\left(Q_c(x,a;\phi_c) - y_c\right)^2\right], \tag{10}$$

$$\mathcal{L}_{V_c}(\psi_c) = \mathbb{E}_{(x,a)\sim\mathcal{D}}[\mathcal{L}_\tau(Q_c(x,a;\phi_c) - V_c(x;\psi_c))]\,. \tag{11}$$

Note the use of the same expectile loss in (11) and (6), but applied to the safety residual $Q_c - V_c$. Here $\tau < 0.5$ causes V_c to track the *lower* quantile of the in-sample safety Q-distribution, yielding a conservative approximation of the feasibility boundary. In implementation we share the same τ hyperparameter across both critics, with opposite sign conventions in expectile regression (i.e., $u > 0$ where $u = Q_c(x,a;\phi_c) - V_c(x;\psi_c)$) for what constitutes a desirable extreme; the reward critic targets the upper quantile while the safety critic targets the lower quantile. The max-backup structure of y_c means that clipped double-Q techniques familiar from reward critics must be applied with a *maximum* operation (i.e., taking the most pessimistic safety estimate): we use two safety Q-networks and set $\bar{V}_c(x') = \max\{Q_c^{(1)}(x',\cdot), Q_c^{(2)}(x',\cdot)\}$, consistently avoiding overoptimistic feasibility estimates at OOD next states.

3.2 Behavior Flow Policy and One-Step Distillation

With critics in place, we turn to policy learning. The central challenge is to produce a policy that (a) stays close to the behavior distribution to avoid distributional shift, (b) is expressive enough to model multimodal and structured action distributions common in robotics datasets, and (c) can be executed at test time with negligible latency. Diffusion-based policies satisfy (a) and (b) through score-matched generative modeling, but their iterative reverse-process sampling incurs $O(T)$ network evaluations per step (Zheng et al., 2024; Wang et al., 2023b). SafeFQL therefore adopts the FQL strategy (Park et al., 2025) of using a flow-matching model as a *fixed behavior teacher* and distilling it into an efficient one-step deployment policy.

Flow behavior teacher. We parameterize the behavior policy π_β via a conditional flow-matching model $\mu_\theta(x,z,t)$, which defines a time-dependent velocity field over actions (Lipman et al., 2023). Given a state $x \sim \mathcal{D}$, a Gaussian sample $z \sim \mathcal{N}(0,I)$, and a time $t \sim \text{Uniform}([0,1])$, the teacher is trained to transport z to the empirical action distribution via the regression objective

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{(x,a)\sim\mathcal{D},\, z\sim\mathcal{N}(0,I),\, t\sim\mathcal{U}([0,1])}\left[\left\|\mu_\theta(x,\, x_t,\, t) - (a - z)\right\|_2^2\right], \tag{12}$$

where $x_t = (1-t)z + ta$ is the straight-line interpolation between the noise sample and the target action. At convergence, integrating the learned velocity field from $t = 0$ to $t = 1$ starting from z generates an action $a \sim \pi_\beta(\cdot|x)$. The flow teacher is trained with behavioral cloning only (no critic signal enters $\mathcal{L}_{\text{flow}}$), which keeps this stage unconditionally stable.

One-step student actor. The deployed policy is a deterministic one-step actor $\mu_\omega(x, z) : \mathcal{X} \times \mathbb{R}^{d_a} \to \mathcal{A}$ that maps a state and a latent noise vector directly to an action, without any iterative integration. To endow the student with the expressiveness of the flow teacher, we define a *distillation loss* that penalizes deviation from the one-step teacher output $\tilde{\mu}_\theta(x, z)$, which is the action produced by running a single integration step of the trained flow model from z conditioned on x:

$$\mathcal{L}_{\text{distill}}(\omega) = \mathbb{E}_{(x,z)\sim\mathcal{D}\times\mathcal{N}(0,I)}\left[\|\mu_\omega(x, z) - \tilde{\mu}_\theta(x, z)\|_2^2\right]. \tag{13}$$

The distillation term serves as a behavior regularizer: it pulls the student actor toward the support of the offline dataset, preventing it from exploiting critic extrapolation errors in regions far from the data (Park et al., 2025). Crucially, the one-step actor μ_ω is the *only* component of SafeFQL that is queried at deployment time, so inference cost is that of a single forward pass through a feedforward network regardless of how many flow steps were used to train the teacher.

3.3 Feasibility-Gated Actor Objective

Given the reward critic Q_r, the safety critic Q_c, and the distillation anchoring from the flow teacher, we now describe how to combine these signals into a well-motivated actor objective. This is the crux of the method, and the design choice here distinguishes SafeFQL from both vanilla FQL and prior soft-constraint offline safe RL approaches.

Limitations of the naive Lagrangian formulation. A natural baseline is to treat the safety constraint as a soft penalty and jointly optimize reward and safety with a Lagrangian multiplier $\eta > 0$:

$$\mathcal{L}^{\text{naive}}_{\text{actor}}(\omega) = \mathbb{E}_{x,z}[-Q_r(x, a_\omega) + \eta\, \max(0, Q_c(x, a_\omega))] + \lambda\, \mathcal{L}_{\text{distill}}(\omega), \tag{14}$$

where $a_\omega = \mu_\omega(x, z)$. The penalty term $\max(0, Q_c)$ is zero when $Q_c < 0$ (predicted feasible) and equal to Q_c when $Q_c \geq 0$ (predicted infeasible). Objective (14) is computationally straightforward since gradients with respect to ω flow through both terms simultaneously, but it has a critical structural flaw, the two terms are commensurate in magnitude and can trade off against each other. Concretely, near the feasibility boundary where Q_c is small but positive, a sufficiently large gradient from $-Q_r$ can dominate and push the actor into the infeasible region. The multiplier η would need to be tuned precisely per task to prevent this, and the right value is not known without online interaction. Empirically, soft-constraint offline methods that rely on this kind of Lagrangian penalty are known to be highly sensitive to the choice of cost limit and multiplier (Zheng et al., 2024; Xu et al., 2022); the coupled optimization of reward, safety, and behavior regularization further exacerbates instability (Lee et al., 2022).

Feasibility-gated objective. SafeFQL replaces the additive tradeoff with an *exclusive-gate* mechanism that enforces strict priority ordering, when the predicted policy action violates feasibility ($Q_c \geq 0$), the actor update completely ignores reward and focuses solely on recovering feasibility; only once the action is predicted feasible ($Q_c < 0$) does the update switch to reward maximization. This is implemented via the binary gate

$$\zeta(x, z) = \mathbb{I}\{Q_c(x, \mu_\omega(x, z)) < 0\}, \tag{15}$$

and the combined actor loss

$$\mathcal{L}_{\text{actor}}(\omega) = \lambda\, \mathcal{L}_{\text{distill}}(\omega) + \mathbb{E}_{(x,z)}\big[\zeta(x, z) \cdot \big(-Q_r(x, a_\omega)\big) + \big(1 - \zeta(x, z)\big) \cdot \max(0, Q_c(x, a_\omega))\big]. \tag{16}$$

Algorithm 1 Safe Flow Q-Learning (SafeFQL)

Require: Offline dataset $\mathcal{D} = \{(x, a, r, \ell, x')\}$, discounts γ, expectile τ, distillation weight λ, calibration parameters $(\epsilon_s, \beta_s, N_{\text{cal}})$

Ensure: Deployed policy μ_ω; corrected safe set $\mathcal{S}_{\delta^*}$

// PHASE 1: CRITIC LEARNING
Initialize Q_r, V_r (reward critics) and Q_c, V_c (safety critics)
for each gradient step **do**
    Update V_r via expectile loss (6); update Q_r via Bellman loss (8)
    Update Q_c via max-backup Bellman loss (10); update V_c via expectile loss (11)
    EMA-update target networks $\bar{V}_r$, $\bar{V}_c$
end for
// PHASE 2: FLOW TEACHER TRAINING
Train behavior flow teacher μ_θ via flow-matching loss (12)
// PHASE 3: FEASIBILITY-GATED ACTOR TRAINING
Initialize one-step actor μ_ω
for each gradient step **do**
    Sample $(x, z) \sim \mathcal{D} \times \mathcal{N}(0, I)$; compute $a_\omega = \mu_\omega(x, z)$
    Compute gate ζ via (15); update μ_ω by minimizing (16)
end for
return μ_ω

The three terms in (16) have distinct and complementary roles. The distillation term $\mathcal{L}_{\text{distill}}$ serves as a universal behavioral anchor, keeping the actor within the support of the offline dataset at all times regardless of the feasibility state. The second term $\zeta \cdot (-Q_r)$ is the reward-maximization signal, which is active only at state-latent pairs where the current actor output is already in the predicted feasible region. The third term $(1-\zeta) \cdot \max(0, Q_c)$ is the feasibility recovery signal, active only when the current output violates the predicted safety boundary, and it pushes the actor in the direction of decreasing Q_c rather than in the direction of reward. Such a feasibility gate ζ eliminates the instability that arises when reward and safety gradients are simultaneously active and point in opposing directions.

In-sample action generation. At each actor update step, the action $a_\omega = \mu_\omega(x, z)$ with $z \sim \mathcal{N}(0, I)$ is sampled fresh, so the stochastic policy π_ω induced by μ_ω is implicitly evaluated at many points per gradient step. No replay buffer of policy actions is needed; the randomness of z provides the necessary coverage of the action distribution to avoid mode collapse under the distillation constraint.

4 Experiments

We evaluate SafeFQL against baseline methods to investigate three critical aspects: (i) its safety rate relative to state-of-the-art constrained offline RL algorithms, (ii) the tradeoff between safety compliance and cumulative reward $\sum_{k=0}^{\infty} r(x_k)$, and (iii) its sampling efficiency during inference compared to prominent alternative generative modeling based frameworks. Our findings confirm that SafeFQL successfully co-optimizes safety and performance, attaining higher safety rates while maintaining competitive reward accumulation.

Baselines: We compare SafeFQL against a diverse set of safety constrained offline reinforcement learning methods. We include **BEAR-Lag** (Lagrangian dual version of Kumar et al. (2019)), **COptiDICE** (Lee et al., 2022), **CPQ** (Xu et al., 2022), **C2IQL** (LIU et al., 2025), **FISOR** (Zheng et al., 2024) and also **SafeIFQL** (Safe Flow Matching version of Kostrikov et al. (2022)). In contrast to these methods, SafeFQL learns a one step policy from offline demonstrations that accounts for future unsafe interactions in advance and therefore, accordingly taking actions that maximize the cumulative reward while staying within the safe region.

Evaluation Metrics: We evaluate all methods based on (i) *safety/cost*, measured as the total number of safety violations incurred before episode termination, and (ii) *performance*, measured via the cumulative episode rewards. These metrics allow us to assess the trade-off between strict safety enforcement and task performance across different offline RL approaches.

4.1 Experimental Case Studies

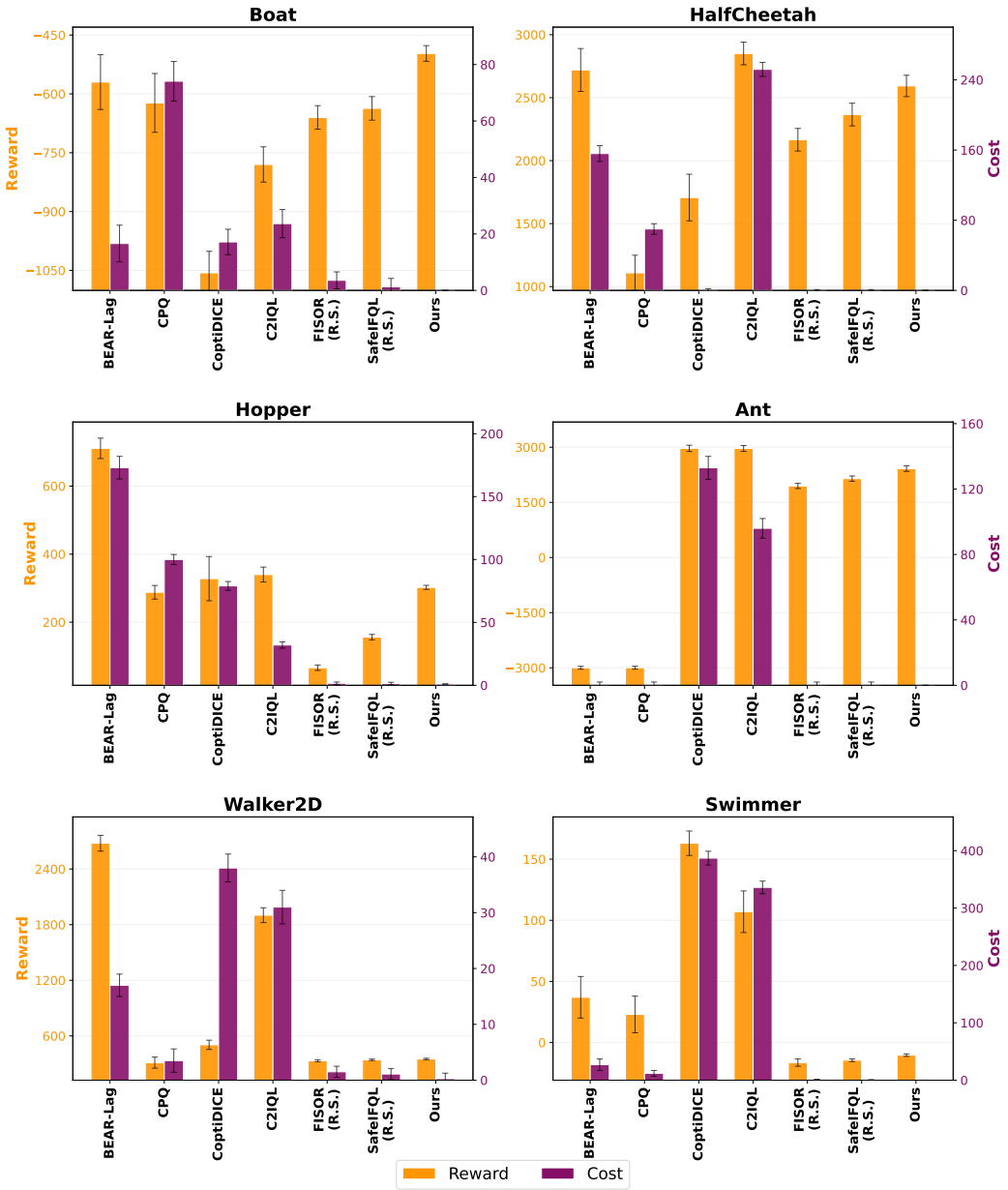

Figure 2: **Evaluation Results.** Reports reward and cost (mean $\pm$ std dev) in all evaluation environments for all frameworks. SafeFQL achieves the lowest costs across all the evaluated environments while achieving highest reward among the frameworks with comparable costs. Some baselines with *(R.S.)* tag represent frameworks that are evaluated using Rejection Sampling (N=16) at evaluation time. Results evaluated across 5 seeds.

For thorough evaluation of our framework against the baselines, we use the following varied set of environments:

- **Safe Boat Navigation:** For our first experiment, we address a two-dimensional collision avoidance problem where a boat, modeled with point mass dynamics, navigates a river (Tayal et al., 2025b). The river's drift velocity changes according to the boat's y-coordinate. The primary goal is to safely bypass obstacles despite this variable drift. We evaluated our approach against baseline methods using fixed 500 randomly selected initial states. Comprehensive details regarding the system dynamics, state space boundaries, and experimental setup can be found in Supplementary Material A.1.
- **Safety Gymnasium:** To further validate our framework, we conducted evaluations within Safety Gymnasium (Ji et al., 2023), specifically focusing on the Safe Velocity suite. For these Safe Velocity tasks, we tested SafeFQL on several high-dimensional MuJoCo environments, including Hopper, Half Cheetah, Swimmer, Walker2D, and Ant. The goal in these settings is to maximize the agent's reward while strictly maintaining its speed below a specified threshold. To ensure a fair comparison against baseline methods, we evaluated performance across a fixed set of 500 randomly sampled initial states for each environment. All the additional details of the framework are available in Supplementary Material A.2 for the readers to refer.

For our experiments within the Safety Gymnasium suite, we employ the standard DSRL dataset for safe offline RL (Liu et al., 2024) while preserving the framework's original reward and safety-violation metrics (Ji et al., 2023). To benchmark our approach against existing baselines, we evaluate performance across a fixed collection of 500 randomly sampled safe initial states for each task.

4.2 Results

While refering the results from Figure 2 for the custom Safe Boat Navigation environment, SafeFQL achieves a significant increase in reward compared to all baselines while maintaining zero violations across all evaluation episodes . This strong performance extends to the high-dimensional Safety Gymnasium tasks (HalfCheetah, Hopper, Ant, Walker2D, and Swimmer), where SafeFQL consistently achieves the lowest safety violations and the highest reward among frameworks with comparable near-zero costs . SafeFQL's success stems from learning a one-step optimal policy that directly outputs high-reward, safe actions . In contrast, baselines optimizing for expected cumulative cost (e.g., BEAR-Lag, CPQ, and COptiDICE) struggle to strictly enforce safety without sacrificing reward , while C2IQL achieves high rewards but incurs inconsistent costs in safety-critical task.

Furthermore, while generative baselines like FISOR and SafeIFQL also account for worst-case safety, they fail to directly learn a single optimal action. Instead, they rely on computationally expensive and suboptimal rejection sampling at inference to filter safe actions from multiple generated candidates. SafeFQL circumvents this entirely by directly outputting the optimal action at each timestep without the need for sampling, establishing it as the most effective and efficient policy among the evaluated frameworks .

4.3 Comparison to Generative Policy Baselines

To further investigate the performance benefits of SafeFQL, we analyzed the safety rate, defined as the percentage of episodes without collisions, in the Figure 3 for the Safe Boat Navigation environment. Generative baseline frameworks like FISOR and SafeIFQL require a large number of action samples (N=16) to achieve safety rates comparable to our method. Because SafeFQL learns a one-shot optimal policy, it successfully outputs the optimal action even when N is restricted to 1.

Besides, we also evaluated the computation time gains of SafeFQL over FISOR and SafeIFQL by analyzing both training and inference times in Figure 4. While SafeFQL requires a longer training compute time compared to the other two frameworks, it more than compensates for this upfront cost with minimal inference latency during deployment. Because FISOR heavily relies on rejection sampling, significant latency is introduced as every time an action must be selected from N candidates

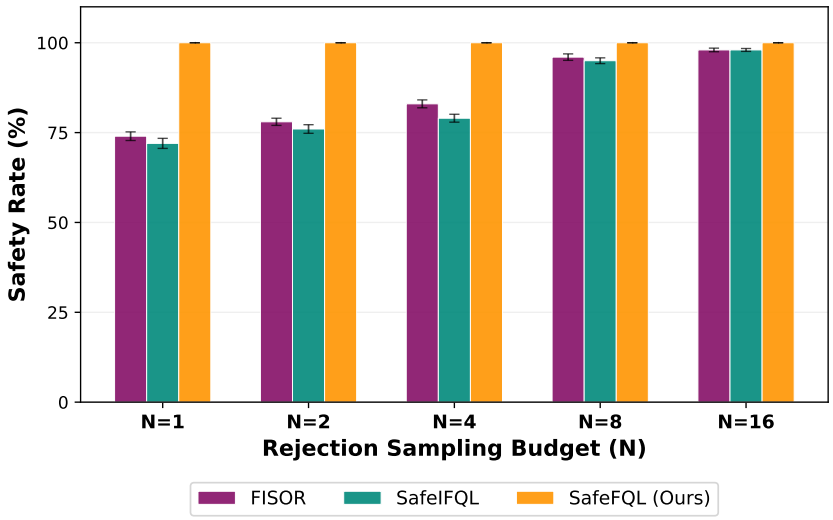

Figure 3: **Action Sampling Efficiency.** Generative policy–based methods (FISOR, SafeIFQL) require rejection sampling to reach high safety (mean ± std dev) across 5 seeds; SafeFQL achieves highest safety in the Safe Boat Navigation environment with only N=1 action sample, while other baselines require larger N.

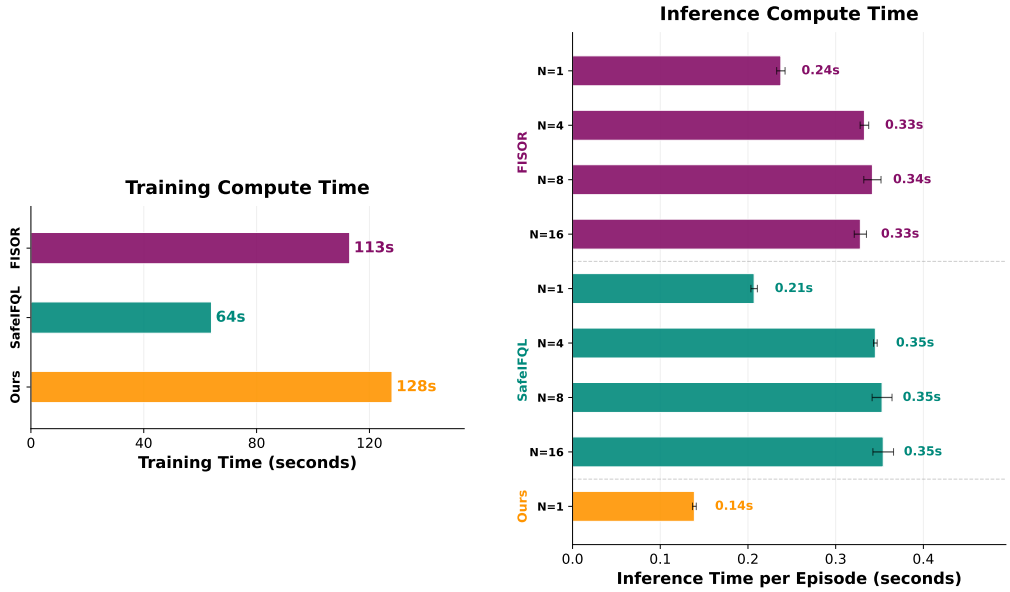

Figure 4: **Computation Time Analysis.** Training Time *(Left)* and Inference Time *(Right)* (mean ± std dev) taken by each of the three generative policy based frameworks when evaluated across 5 independent trial runs.

based on cost and reward Q-values. Even when N is set to 1 for FISOR and SafeIFQL and rejection sampling is disabled, inference time remains high due to the multi-step denoising process inherent to both DDPM and Flow Matching policies. This demonstrates that trading a one-time higher training cost for SafeFQL yields a vastly more efficient, highly accurate, single-step policy. Ultimately, this highlights the immense practical effectiveness of SafeFQL for high-frequency, real-time control loops once the model is trained.

5 Conclusion, Limitations and Future works

We introduced Safe Flow Q-Learning (SafeFQL), a scalable offline safe reinforcement learning framework that synthesizes the expressivity of generative flow models with the rigorous safety principles of Hamilton-Jacobi reachability. By distilling a multi-step flow policy into an efficient one-step actor, SafeFQL eliminates the prohibitive computational costs of iterative action sampling at deployment, achieving an inference time speedup of $2.5\times$. Empirically, SafeFQL maintains competitive rewards while achieving near-zero violations using a single action proposal across both high-dimensional Safety Gymnasium and custom navigation tasks.

While SafeFQL demonstrates robust empirical safety, we identify some scope of algorithmic refinement. Currently we use hard indicator mask for the use of Q-critic functions in Algorithm 1 which could theoretically yield a non-smooth loss landscape and might impact training at times. Therefore, one could explore use of continuous masking functions or soft Lagrangian relaxations to improve framework stability, however, that requires hyperparameter finetuning, which can be undesirable.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 22–31. JMLR.org, 2017.

Marvin Alles, Nutan Chen, Patrick van der Smagt, and Botond Cseke. Flowq: Energy-guided flow policies for offline reinforcement learning, 2025. URL https://arxiv.org/abs/2505.14139.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. DOI: 10.1609/aaai.v32i1.11797. URL https://ojs.aaai.org/index.php/AAAI/article/view/11797.

Eitan Altman. *Constrained Markov decision processes*. Routledge, 2021.

Aaron D Ames, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control*, pp. 6271–6278. IEEE, 2014.

Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8): 3861–3876, 2017. DOI: 10.1109/TAC.2016.2638961.

Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2242–2253. IEEE, 2017.

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.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

Yusuf Umut Ciftci, Darren Chiu, Zeyuan Feng, Gaurav S Sukhatme, and Somil Bansal. Safe-gil: Safety guided imitation learning for robotic systems. *arXiv preprint arXiv:2404.05249*, 2024.

Jaime F. Fisac, Neil F. Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J. Tomlin. Bridging hamilton-jacobi safety analysis and reinforcement learning. In *2019 International*

Conference on Robotics and Automation (ICRA), pp. 8550–8556, 2019. DOI: 10.1109/ICRA.2019.8794107.

Scott Fujimoto and Shixiang Gu. A minimalist approach to offline reinforcement learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Q32U7dzWXpc.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

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

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety gymnasium: A unified safe reinforcement learning benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=WZmlxIuIGR.

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

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf.

Jongmin Lee, Cosmin Paduraru, Daniel J Mankowitz, Nicolas Heess, Doina Precup, Kee-Eung Kim, and Arthur Guez. COptiDICE: Offline constrained reinforcement learning via stationary distribution correction estimation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=FLA55mBee6Q.

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.

Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. When data geometry meets deep function: Generalizing offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Qian Lin, Bo Tang, Zifan Wu, Chao Yu, Shangqin Mao, Qianlong Xie, Xingxing Wang, and Dong Wang. Safe offline reinforcement learning with real-time budget constraints. In *International Conference on Machine Learning*, 2023.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Zifan LIU, Xinran Li, and Jun Zhang. C2IQL: Constraint-conditioned implicit q-learning for safe offline reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=97N3XNtFwy.

Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. In *International Conference on Machine Learning*, 2023.

Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, and Ding Zhao. Datasets and benchmarks for offline safe reinforcement learning. *Journal of Data-centric Machine Learning Research*, 2024.

Ian M. Mitchell. A toolbox of level set methods. In *A Toolbox of Level Set Methods*, 2005. URL https://api.semanticscholar.org/CorpusID:59892255.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline RL? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=nyp59a31Ju.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning (ICML)*, 2025.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pp. 745–750, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937933. DOI: 10.1145/1273496.1273590. URL https://doi.org/10.1145/1273496.1273590.

Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation, 2024.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID lagrangian methods. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9133–9143. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/stooke20a.html.

Manan Tayal and Mumuksh Tayal. Epigraph-guided flow matching for safe and performant offline reinforcement learning. *arXiv preprint arXiv:2602.08054*, 2026.

Manan Tayal, Aditya Singh, Shishir Kolathaya, and Somil Bansal. A physics-informed machine learning framework for safe and optimal control of autonomous systems. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=SrfwiloGQF.

Mumuksh Tayal, Manan Tayal, Aditya Singh, Shishir Kolathaya, and Ravi Prakash. V-ocbf: Learning safety filters from offline data via value-guided offline control barrier functions. *arXiv preprint arXiv:2512.10822*, 2025b.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2018.

Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments. In *International Conference on Machine Learning*, pp. 36593–36604. PMLR, 2023a.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *The Eleventh International Conference on Learning Representations*, 2023b.

Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8753–8760, 2022.

Jiazhi Zhang, Yuhu Cheng, C.L. Philip Chen, Hengrui Zhang, and Xuesong Wang. Diffusion policy distillation for offline reinforcement learning. *Neural Networks*, 190:107694, 2025a. ISSN 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2025.107694.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=HA0oLUvuGI.

Weiye Zhao, Tairan He, Rui Chen, Tianhao Wei, and Changliu Liu. Safe reinforcement learning: A survey. *arXiv preprint arXiv:2302.03122*, 2023.

Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=cM95sT3gM3.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Description of the Experiments

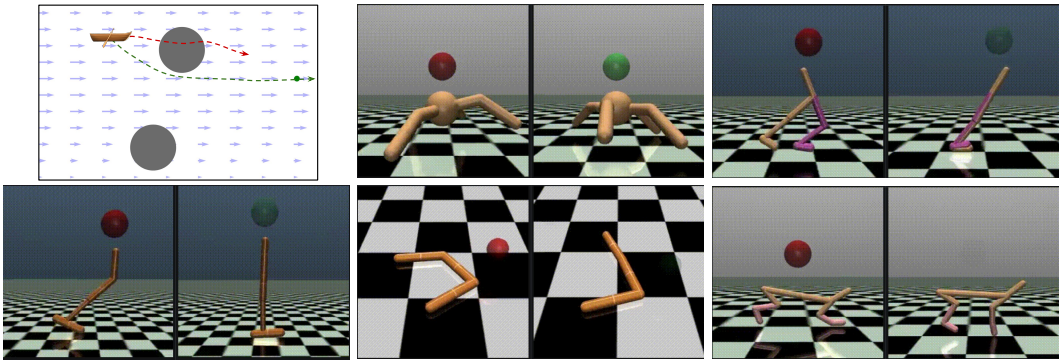

Figure 5: **Illustration of Evaluation Environments.** *(TopLeft):* Environment depicts the Safe Boat Navigation task with 2 obstacles and a goal point in a drifting river. *(Remaining):* Environments are the standard Safety Gymnasium environments (Ji et al., 2023) from its Safe Velocity suite.

A.1 Boat Navigation

The 2D Boat state is $x \in \mathcal{X} = [-3, 2] \times [-2, 2]$, with $x = [x_1, x_2]^\top$ representing the boat's Cartesian coordinates (x_1, x_2). We use a dense, distance-based step reward that encourages progress toward a fixed goal:

$$r(x) \;=\; C \cdot \Big(- \big\| [x_1, x_2]^\top - [x_{g1}, x_{g2}]^\top \big\| \Big), \tag{17}$$

where the goal is $[x_{g1}, x_{g2}]^\top = [0.5, 0.0]^\top$ and $C = 0.1$. Maximizing $r(x)$ therefore drives the boat toward the goal location.

The discrete-time dynamics are given by

$$\begin{aligned} x_{1,t+1} &= x_{1,t} + \big(a_{1,t} + 2 - 0.5\, x_{2,t}^2\big)\, \Delta t, \\ x_{2,t+1} &= x_{2,t} + a_{2,t}\, \Delta t, \end{aligned}$$

where Δt is the integration timestep, $(a_{1,t}, a_{2,t})$ are the control inputs satisfying $a_{1,t}^2 + a_{2,t}^2 \le 1$, and the term $2 - 0.5 x_{2,t}^2$ models the state-dependent longitudinal drift along the x_1-axis.

Obstacles and the failure region are encoded via the safety function $\ell(x)$. We define $\ell(x)$ using the negative of signed distance (plus the obstacle radius) to two circular obstacles:

$$\ell(x) \;:=\; \max\big(0.4 - \|x - [-0.5,\, 0.5]^\top\|,\; 0.5 - \|x - [-1.0,\, -1.2]^\top\|\big). \tag{18}$$

By this definition, $\ell(x) > 0$ indicates that the boat lies inside an obstacle; the super-level set $\{x : \ell(x) > 0\}$ therefore defines the failure region.

Offline data generation. Because this environment is custom, we construct an offline dataset for training and evaluation. We sample 2,500 initial states uniformly from $\mathcal{X}$ and simulate each trajectory for 400 discrete timesteps with step size $\Delta t = 0.005$ s. During data collection control inputs are sampled uniformly at random from the admissible action set (subject to the norm constraint), ensuring a wide variety of state–action coverage for downstream learning of safe controllers.

A.2 Safety MuJoCo Environments

To evaluate performance on higher-dimensional systems we use the MuJoCo-based Safety Gymnasium environments Ji et al. (2023). These environments implement *safe velocity* tasks in which the agent incurs a cost whenever its instantaneous velocity exceeds a task-specific threshold.

At each step the environment provides a binary cost defined by the velocity constraint:

$$\text{cost}_t \;=\; \mathbb{I}\big[V_{\text{current},t} > V_{\text{threshold}}\big], \tag{19}$$

where $\mathbb{I}[\cdot]$ is the indicator function. Equivalently, we express safety with the continuous safety function

$$\ell(x) \;=\; V_{\text{current}}(x) - V_{\text{threshold}}, \tag{20}$$

so that $\ell(x) \le 0$ corresponds to the safe set. Using $\ell(x)$ provides a dense, continuous safety signal rather than a sparse $\{0, 1\}$ cost.

The per-environment threshold velocities $V_{\text{threshold}}$, integration timesteps Δt, and action-space bounds $\mathbb{U}$ are taken from the official documentation; the values we used are summarized in Table 1.

Table 1: Velocity thresholds $V_{\text{threshold}}$, timestep Δt, and action-space $\mathbb{U}$ for the MuJoCo Safety Gymnasium environments used in our experiments (values from the official docs).

Environment	**Hopper**	**HalfCheetah**	**Ant**	**Walker2D**	**Swimmer**
$V_{\text{threshold}}$	0.7402	3.2096	2.6222	2.3415	0.2282
Δt (s)	0.008	0.05	0.05	0.008	0.04
$\mathbb{U}$	$[-1, 1]^3$	$[-1, 1]^6$	$[-1, 1]^8$	$[-1, 1]^6$	$[-1, 1]^2$

B Additional Results

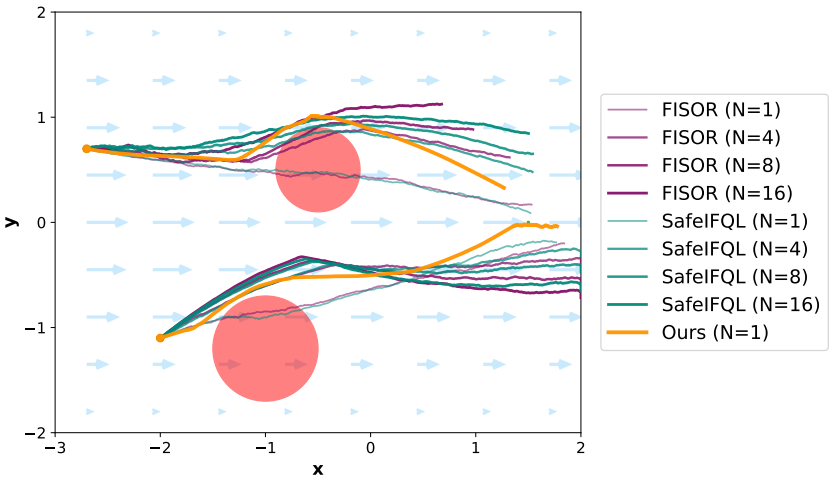

Figure 6: **Boat Navigation Environment Trajectory Rollouts** for Generative Policy based Frameworks when different candidate action pool sizes (represented by N) are used. For this plot, we use $N \in \{1, 4, 8, 16\}$ for FISOR and SafeIFQL frameworks. For SafeFQL, we stick to N=1 as the framework doesn't require action rejection sampling.

To Visually compare action sample efficiency for the generative policy based methods (FISOR, SafeIFQL and SafeFQL (Ours)), the trajectories across different sample sizes (Figure 6) highlights the difference. SafeFQL not only achieves zero collisions across its trajectories, but it also reaches

closest to the goal among all safe trajectories by the baselines. Furthermore, because SafeFQL circumvents the need for rejection sampling, it preserves its mathematically optimal nature while completely avoiding the computational hassle of multi-action sampling at inference time.

C Experimental Details

C.1 Experimental Hardware

To ensure a fair comparison, all experiments were performed on the same system with a 14th-Gen Intel Core i9-14900KS CPU with 64 GB of RAM and an NVIDIA GeForce RTX 5090 GPU, used for both training and evaluation.

C.2 Network Architecture and Training Details of the Proposed Algorithm

We have compiled and listed down all the hyperparameters that we used to perform our experiments and report the results. These training settings for all the environments are detailed in the Table 2. For the MuJoCo environments, we use the widely accepted DSRL Liu et al. (2024) dataset.

During the training, we set the τ for expectile regression in section 3 to 0.9. And we use clipped double Q-learning (Fujimoto et al., 2018) for both reward and safety Q-critic functions, taking a minimum of the two Q values. We update the target Q networks using Exponential Moving Average (EMA) where the weight to the new parameters is set to 0.005. Following Kostrikov et al. (2022), we clip exponential advantages to $(-\infty, 100]$ in feasible part and $(-\infty, 150]$ in infeasible part. For our paper, we used Flow Q-Learning implementation from Park et al. (2025).

C.3 Hyperparameters for the Baselines

Hyperparameters for the remaining safe offline-RL baselines (COptiDICE, BEAR-Lag, CPQ, C2IQL, FISOR) are listed in Table 3. We employ the official implementations of BEAR-Lag, CPQ and COptiDICE from Liu et al. (2024), C2IQL from LIU et al. (2025), and FISOR from Zheng et al. (2024). For SafeIFQL we follow the hyperparameter choices of Zheng et al. (2024) however, Flow-Matching implementation for its IFQL component is adapted from Park et al. (2025). All methods are trained on the same DSRL datasets.

Table 2: Hyperparameters for the Algorithm (SafeFQL).

Hyperparameter	**Value**
Network Architecture	Multi-Layer Perceptron (MLP)
Activation Function	ReLU
Optimizer	Adam optimizer
Learning Rate	3×10^{-4}
Discount Factor (γ)	0.99
Time Step Intervals (FM Policy)	10
Boat Navigation	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	2
Number of Hidden Layers (π)	3
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	1M
Safe Velocity Hopper	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	3
Number of Hidden Layers (π)	4
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	1.32M
Safe Velocity Half-Cheetah	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	3
Number of Hidden Layers (π)	4
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	249K
Safe Velocity Ant	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	3
Number of Hidden Layers (π)	4
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	2.09M
Safe Velocity Walker2D	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	3
Number of Hidden Layers (π)	4
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	2.12M
Safe Velocity Swimmer	
Number of Hidden Layers (Q_r, V_r, Q_c, V_c)	3
Number of Hidden Layers (π)	4
Hidden Layer Size (Both)	256 neurons per layer
Dataset Size	1.68M

Table 3: Detailed Hyperparameters for Baseline Special Networks. (Layers, Units) notation refers to hidden layers and units per layer. All baselines utilize the DSRL dataset standards Liu et al. (2024).

Baseline	Network Component	Architecture Specification
C2IQL (LIU et al., 2025)	Actor / Critic / Value Nets	MLP, (256, 2)
	Cost Reconstruction Model	**MLP, (5 layers, 512 units each)**
	Reward / Cost Advantage	MLP, (256, 2)
CPQ (Xu et al., 2022)	Actor (Policy) Net	MLP, (256, 2)
	Constraint-Penalized Q	**Ensemble DoubleQCritic, (256, 2)**
	Cost Critic Net	MLP, (256, 2)
FISOR (Zheng et al., 2024)	Diffusion Denoiser (Actor)	DiffusionDenoiserMLP, (256, 2)
	Feasibility Classifier	**MLP, (256, 2), Sigmoid Output**
	Energy/Value Guidance	MLP, (256, 2)
COptiDICE (Lee et al., 2022)	**Dual / ν Network**	**DualNet (Value-like), (256, 2)**
	Actor (Extraction) Net	MLP, (256, 2)
BEAR-Lag (Lag. dual of (Kumar et al., 2019))	**VAE (Support Model)**	**MLP, (750, 2)**
	Actor (Policy) Net	SquashedGaussianMLPActor, (256, 2)
	Cost / Reward Critics	MLP, (256, 2)