Fast & Stable Control of Coupled Solid-Fluid Dynamic Systems

JIE CHEN, Nankai University, China
ZHERONG PAN, LIGHTSPEED, USA
BO REN*, Nankai University, China

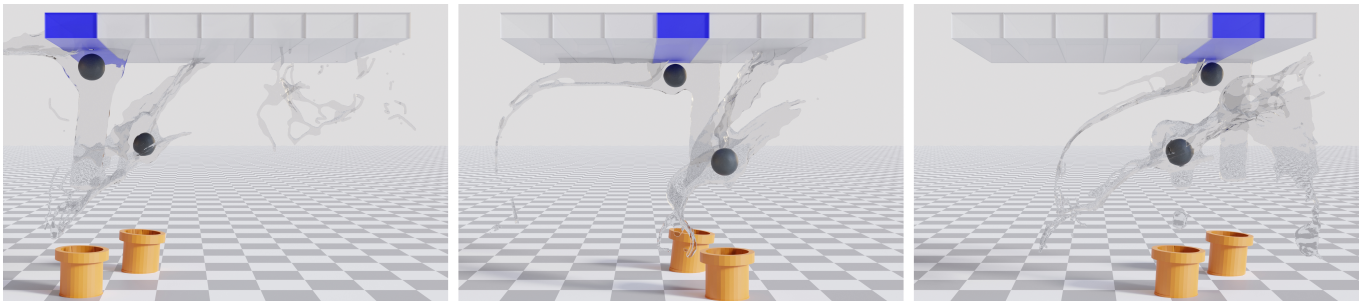

Fig. 1. Our off-policy reinforcement learning framework for complex fluid-solid coupling control achieves stable and reliable results through efficient value estimation and policy guiding. In this benchmark, we achieve stable multi-goal control in the double solid music player task over long horizons. Our trained policy successfully controls the two fluid drivers (yellow) to prevent the balls from falling, hit the goal key, and play the music at various tempos.

We propose a Reinforcement Learning (RL) algorithm that combines several novel techniques to achieve more stable and robust control results for coupled solid-fluid systems. Our method utilizes the twin-delayed actor-critic algorithm to efficiently utilize off-policy data and achieve faster convergence. For more accurate estimations of the value function to guide the search of optimal policies, we use the Boltzmann softmax operator to reduce the bias of estimation. We further introduce a novel two-step Q-value estimator to reduce the well-known under-estimation issue. Finally, to mitigate the requirement of excessive exploration under sparse rewards, we propose the Fluid Effective Domain Guidance (FEDG) algorithm to guide policy exploration, where the policy for an easier task is trained jointly with that for a harder task. Put together, our framework achieves state-of-the-art performance in complex fluid-solid coupling control benchmarks, delivering stable and reliable performance in both 2D and 3D tasks over long horizons.

CCS Concepts: • **Computing methodologies** → **Physical simulation**.

Additional Key Words and Phrases: Fluid-solid coupling, optimal control, reinforcement learning

ACM Reference Format:
Jie Chen, Zherong Pan, and Bo Ren. 2025. Fast & Stable Control of Coupled Solid-Fluid Dynamic Systems. In *SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25), December 15–18, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3757377.3763997

*Corresponding author

Authors' Contact Information: Jie Chen, VCIP, College of Computer Science, Nankai University, Tianjin, China, jiechen@mail.nankai.edu.cn; Zherong Pan, LIGHTSPEED, Lacey, Washington, USA, zherong.pan.usa@gmail.com; Bo Ren, VCIP, College of Computer Science, Nankai University, Tianjin, China, rb@nankai.edu.cn.

SA Conference Papers '25, Hong Kong, Hong Kong

ACM ISBN 979-8-4007-2137-3/25/12
https://doi.org/10.1145/3757377.3763997

1 INTRODUCTION

Physics-based animation is an active area of research and reaches a high level of maturity. However, designers still suffer from trial and error in fine-tuning the setup of physical scenarios to generate animations that produce the desired visual effects. Therefore, to further automate the animation design pipeline, a row of animation control algorithms has been proposed, in which successful examples involve character control [Bergamin et al. 2019; Chen et al. 2024b; Cho et al. 2021], dance generation [Alexanderson et al. 2023; Chen et al. 2021; Tseng et al. 2023], and the control of fluid and solid-fluid coupled systems [Chu et al. 2021; McNamara et al. 2004]. Among these works, the control of fluid, particularly solid-fluid coupled systems, is challenging mainly due to a high degree of freedom (DOF) and the intricate fluid motions that are highly complex to predict and even harder to modify in a user-desired manner. The control of underactuated coupled solid-fluid dynamic systems is crucial and complex for applications like underwater and aerial robots. In this work, we focus on addressing the challenge in such systems. Although interest has been growing among researchers in computer graphics, physics, and mechanical engineering [Wang et al. 2023], practical control frameworks remain scarce in the literature.

Recently, a row of novel methods is proposed to tame the solid-fluid control problem. A line of research [Holl et al. 2020a; Li et al. 2024, 2023; Takahashi et al. 2021] develops differentiable fluid simulators that employ the gradient to optimize for desirable animations, which is advantageous for tasks that adhere to the differentiable assumptions. However, these methods require specialized simulators and may not be applicable to tasks that involve non-smooth behaviors such as interfacial change between the two coupling objects. RL, on the other hand, provides greater flexibility by working with any simulator and handling a diverse range of control tasks. Ma

et al. [2018] are the first to adopt the on-policy RL [Schulman et al. 2015] in fluid-solid coupling tasks and employ the autoencoder for fluid velocity field feature extraction. Then, Ren et al. [2022] apply an off-policy method, significantly improving data efficiency, and employ meta-learning [Rakelly et al. 2019] to achieve generalization across different simulator parameters. These studies validate the feasibility of RL in fluid-solid coupling tasks. However, their policies remain stable for only a short time period and are suboptimal for complex tasks. To achieve more stable and reliable control performance under long horizons, numerous challenges need to be addressed in the RL algorithms, which require urgent attention.

We argue that there are two major issues preventing the prior work [Ren et al. 2022] from efficiently finding a stable controller. First, it is widely known that the vanilla actor-critic RL suffers from the estimation bias in value functions [Fujimoto et al. 2018; Pan et al. 2020]. As such bias is propagated over long trajectories, the policy performance degrades with longer horizons. Some works [Haarnoja et al. 2018; Hasselt 2010] mitigate the over-estimation, which in turn leads to under-estimation. However, neglecting the under-estimation bias can lead to suboptimal policies. As a second issue, in coupled control tasks, agents are typically learning under sparse rewards which require extensive exploration to feel the impact of reward signals, leading to slow learning and potentially suboptimal outcomes, or even complete failure to converge.

This paper proposes a series of modifications that improve the convergence of controller learning. Based on the twin-delayed actor-critic algorithm, which already deals with over-estimations but suffers from under-estimation, we introduce the Boltzmann softmax operator to significantly reduce the estimation bias in value functions. Further, we introduce a two-step Bellman operator to effectively mitigate under-estimation. Finally, we propose the Fluid Effective Domain Guidance (FEDG) algorithm to tackle sparse rewards. FEDG co-trains two policies with shared architecture, one for a simpler low-level task and the other for a harder high-level task. Our contributions are summarized below:

- An off-policy RL algorithm for fast & stable controller optimization applied to solid-fluid coupled systems.
- The FEDG for bootstrapping policy under sparse rewards.
- An open-source coupled solid-fluid control system.

Put together, through a row of 2D and 3D coupled control tasks, we show that our method has improved convergence and more stable controller performance over long horizons as illustrated in figure 1. The source code is publicly available at https://github.com/lvsichan/FluidControl2025.

2 RELATED WORK

In this section, we review the related work on fluid control and advancements in RL with a focus on estimation bias and exploration.

Fluid control methods can be divided into appearance control and coupled-rigid-body control. The primary objective of appearance control is to allow fluids to naturally and accurately flow into user-specified shapes. Early researchers employed external force control to physically warp fluid density fields into a series of key-frame shapes [McNamara et al. 2004; Treuille et al. 2003] or a single target shape [Fattal and Lischinski 2004; Shi and Yu 2005]. Thürey et al. [2009] preserves small-scale details by applying control forces only to coarse velocity components. Nielsen and Bridson [2011] aligns high-resolution simulations with low-resolution versions by restricting the solve to a thin outer shell, improving speed and compatibility with standard fluid simulators. Pan and Manocha [2017] controls smoke animations by optimizing control force fields to match key-frames, significantly improving speed over previous methods. Chu et al. [2021] proposes a data-driven conditional adversarial model, enabling control through obstacles, physical parameters, kinetic energy, and vorticity. Tang et al. [2023] balances deformations and physical properties using CNNs and a differentiable simulator, achieving accurate and visually appealing results. Chen et al. [2024a] utilizes Laplacian Eigenfluids and the adjoint method, enabling efficient real-time simulation, editing, and control.

Coupled-rigid-body control refers to directly or indirectly utilizing fluid to drive (rigid) objects within the system to achieve the desired state of motion. Ma et al. [2018] employs RL to control a 2D coupling system by applying boundary forces, realizing physically plausible animations. To control complex physical systems over a long horizon, Holl et al. [2020b] splits planning and control, using a predictor and control network trained together with a differentiable PDE solver. Combining meta-RL and a novel task representation, Ren et al. [2022] designs a learning-based controller for fluid-solid coupling systems that adapts to changing dynamics and tasks without retraining. Ramos et al. [2022] proposed to use a differentiable simulator and physically interpretable loss terms to train controllers that generalize well to new conditions. Xian et al. [2023] proposes a simulation platform with a differentiable physics engine that addresses challenges in robotic fluid manipulation through domain-specific optimization schemes. Li et al. [2023] designs a differentiable SPH-based fluid-rigid coupling simulator that tackles gradient instability and high computational cost.

Reinforcement learning algorithms are often categorised as either on-policy or off-policy [Sutton and Barto 1998], depending on whether the training data is collected by the current learning policy. On-policy methods learn exclusively from data generated by the current policy. By contrast, off-policy methods can reuse past experiences collected by any previous policy, greatly improving the data efficiency. On-policy methods like PPO [Schulman et al. 2017] are stable but slow, making them suitable when data collection is inexpensive but less ideal for computationally expensive scenarios such as fluid simulation. In contrast, off-policy methods are more sample-efficient, enabling faster convergence with less data, but they often suffer from instability due to estimation bias.

Estimation bias is a ubiquitous challenge in RL, where initial estimation errors can accumulate over successive timesteps, leading to substantial biases that may degrade agent performance or impede algorithm convergence. Thrun and Schwartz [1993] highlights that the max operator can lead to over-estimation in Q-learning [Watkins and Dayan 1992]. Hasselt [2010] then introduces double Q-learning which eliminates over-estimation but again introduces under-estimation. DDPG [Lillicrap et al. 2016], a preeminent algorithm in the domain of continuous controls is also subject to estimation bias. In response, Fujimoto et al. [2018] introduces TD3, which utilizes dual estimators for the critic, employing the minimum value from two Q-networks to avoid over-estimation.

While the aforementioned methods have successfully mitigated over-estimation bias, under-estimation bias still remains, which can adversely affect overall performance [Ciosek et al. 2019] and is the main technical challenge that we address.

Exploration to discover high-reward regions in the state space is crucial in RL. This is notably more challenging in continuous control environments, particularly in fluid control tasks, where the action space exhibits a pronounced increase in complexity. The simplest strategy is to randomly perturb the actions themselves. Stochastic policies, such as SAC [Haarnoja et al. 2018], naturally incorporate randomness through action sampling. Deterministic policies, such as TD3 [Fujimoto et al. 2018] and SD3 [Pan et al. 2020], enhance exploration by adding random noise such as pink noise [Eberhard et al. 2023] to actions. However, these techniques can waste computation by exploring in unimportant low-reward areas [Lee et al. 2021]. In contrast, Luo et al. [2023] proposes Self-Guided Exploration Strategy (SGES) for complex sequential tasks, employing simpler learned low-level sub-task policies to guide the exploration of a more complex high-level policy.

3 PRELIMINARIES

In this section, we formulate our problem of coupled solid-fluid control and then introduce the main idea behind off-policy RL, which is adopted as the backbone of our main algorithm framework.

3.1 Problem Statement

Our solid-fluid coupled system comprises three main components: the fluid driver, the fluid, and the target solid. The moving least squares material point method (MLS-MPM) [Hu et al. 2018] is primarily used as the simulation algorithm and shape matching [Müller et al. 2005] is employed for the target solid to prevent deformation.

The problem of controlling the solid-fluid coupled system, under the RL paradigm, can be effectively formulated by a Markov Decision Process (MDP) [van Otterlo and Wiering 2012] as defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the set of all states, $\mathcal{A}$ the set of all actions, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, $\mathcal{P}$ the state transition probability, and γ the discount factor. We take the assumption that the action space $\mathcal{A}$ is bounded. At each time t, the agent observes a state $s \in \mathcal{S}$ and selects an action $a \in \mathcal{A}$ according to its policy $\pi : \mathcal{S} \to \mathcal{A}$. The environment then transitions to the next state s_{t+1} and yields a reward r_{t+1}. The goal of policy $\pi(\cdot;\phi)$ parameterized by ϕ is to maximize the long-term cumulative discount rewards:

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, a_t \sim \pi(s_t;\phi), s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)\right].$$

To ensure the seamless integration of our control policies into other fluid simulation algorithms, such as fluid-implicit-particle (FLIP) method [Zhu and Bridson 2005] and smoothed particle hydrodynamics (SPH) methods [Becker and Teschner 2007; Müller et al. 2003], we use a general-purpose state representation. Specifically, the coupled system state $s \in \mathcal{S}$ for RL agents comprises three components: $s \triangleq \begin{pmatrix} d & q & u \end{pmatrix}$. d is the state of the fluid driver and q is the state of the target solid, e.g. position, orientation, velocity and so on. Finally, u is the velocity field feature of the fluid, extracted by the pretrained autoencoder [Vincent et al. 2008] as in [Ma et al. 2018; Ren et al. 2022]. The reward function $\mathcal{R}$ generally uses the state q_t at time t and the desired state q_d of the target solid to calculate the reward r_t. $\mathcal{R}$ varies across different control tasks and will be specifically defined. In our work, the RL policy $\pi(\cdot;\phi)$ aims to compute control actions a_t for the fluid driver to manipulate the fluid behavior such that the target solid achieves its desired state q_d, while maximizing the expected discounted return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$.

3.2 Actor-Critic RL with Deterministic Policy Gradient

We employ an off-policy actor-critic framework with deterministic policy gradient [Silver et al. 2014] for policy optimization. The actor utilizes deterministic policy gradient optimization, guided by the critic's value function estimation, to select actions that maximize expected long-term returns. In such methods, the critic's accurate estimation of either the state-value or action-value function plays a decisive role in both the convergence properties and ultimate performance of the policy. In our work, the policy $\pi : \mathcal{S} \to \mathcal{A}$, parameterized by ϕ, maps states $s \in \mathcal{S}$ to actions $a = \pi(s;\phi) \in \mathcal{A}$. The critic evaluates action advantages through a parameterized state-action value function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ with parameters θ:

$$Q(s, a; \theta) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right]. \tag{1}$$

The Q-function estimates the maximum cumulative reward achievable by taking action a in state s. Suppose our policy is optimal, denoted as $\pi^\star(s) = \arg\max_a Q^\star(s, a, \theta)$, then the corresponding state-action value function is denoted as $Q^\star : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which satisfies the Bellman optimality equation:

$$Q^\star(s, a) = \mathbb{E}\left[r(s, a) + \gamma \max_{a'} Q^\star(s', \pi^\star(s')) \,\middle|\, s' \sim \mathcal{P}\right].$$

In order to learn the optimal policy and corresponding value function, off-policy RL uses two steps starting from an initial guess of ϕ and θ. First, the temporal difference error is minimized to approximate the optimal state-action value function through the Bellman residual loss:

$$\mathcal{L}_{\text{critic}} = \mathbb{E}\left[\|\mathcal{B}(s, a, s', a') - Q(s, a; \theta)\|^2 \,\middle|\, (s, a, s') \sim \mathcal{D}\right], \tag{2}$$

where $\mathcal{D}$ is the experience replay buffer, $a' = \pi(s';\phi)$. $\mathcal{B}$ is the Bellman operator which takes the following form in the standard setting [Lillicrap et al. 2016]:

$$\mathcal{B}(s, a, s', a') = r(s, a) + \gamma Q(s', a'; \theta), \tag{3}$$

where $\langle s, a, s', a' \rangle$ is an unrolled partial trajectory over two timesteps. Minimizing the $\mathcal{L}_{\text{critic}}$ leads to more accurate Q-function estimates. Then, we can update $\pi(s;\phi)$ by the deterministic policy gradient:

$$\nabla_\phi J(\pi(\cdot;\phi)) = \mathbb{E}\left[\nabla_\phi(\pi(s;\phi))\nabla_a(Q(s, a; \theta))\,|_{a=\pi(s;\phi)} \,\middle|\, s \sim \mathcal{D}\right] \tag{4}$$

The gradient of the Q-function with respect to the action guides policy updates toward higher expected return. A limitation in this optimization paradigm stems from the estimation bias in $Q(s, a; \theta)$ during policy improvement iterations, which is observed and analyzed in previous work [Fujimoto et al. 2018; Pan et al. 2020], which also serves as the focal point of this work.

4 METHOD

In this section, we introduce a series of enhancements to the actor-critic RL that significantly improve its performance in the solid-fluid coupled control task. We first adopt a Boltzmann softmax operator based on the clipped Q-value estimator, which reduces the over-estimation and the variance. Next, we introduce a novel two-step

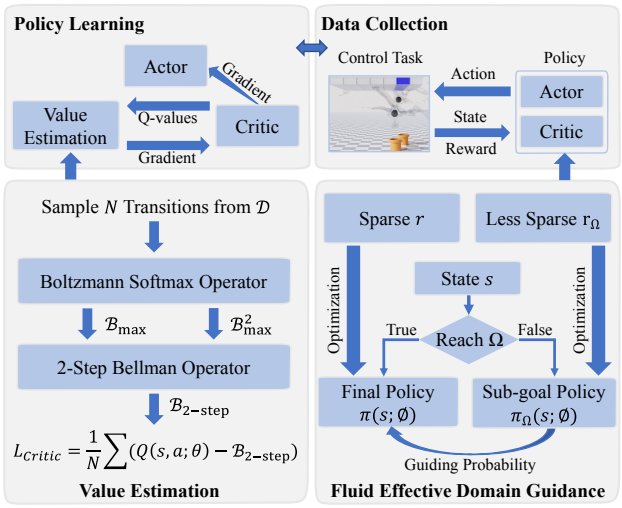

Fig. 2. Architecture of our off-policy RL framework for solid-fluid coupled control tasks. The off-policy reinforcement learning framework efficiently balance the exploration and exploitation using off-policy transition dataset (top row). Our critic loss integrates the Boltzmann softmax operator and the two-step Bellman operator to tackle the estimation bias problem for Q-values (bottom left). The Fluid Effective Domain Guidance (FEDG) guides the exploration of the high-level policy optimizing a sparse reward r through an auxiliary policy optimizing a less sparse reward r_Ω (bottom right).

Bellman operator to further mitigate under-estimation. Finally, we introduce the FEDG algorithm for guiding policies under sparse rewards. The pipeline of our method is highlighted in figure 2.

Boltzmann Softmax Operator. It is known that the max operator in equation 3 can lead to over-estimation [Hasselt et al. 2016], which is mitigated in TD3 [Fujimoto et al. 2018] using two value estimators parameterized by θ_1 and θ_2, respectively. Specifically, TD3 introduces the following clipped double-Q Bellman operator:

$$\mathcal{B}_{\text{TD3}}(s,a,s',a') = r(s,a) + \gamma Q_{\min}(s',a')$$
$$Q_{\min}(s',a') = \min_{i=1,2} Q(s',a';\theta_i).$$

Although $\mathcal{B}_{\text{TD3}}$ mitigates over-estimation, it can still suffer from a high variance because it is essentially a sample approximation of the groundtruth Bellman operator (see equation 3), where the Q-value of the next state is maximized over all actions. Although such maximization does not have a closed-form solution, we could use the Boltzmann softmax operator to approximate such maximization by sampling [Pan et al. 2020], where is defined as follows:

$$\text{Softmax-Q}_\beta(s) = \int_{a\in\mathcal{A}} \frac{\exp(\beta Q_{\min}(s,a))Q_{\min}(s,a)}{\int_{a'\in\mathcal{A}}\exp(\beta Q_{\min}(s,a'))da'}da$$
$$\mathcal{B}_{\max}(s,a,s') = r(s,a) + \gamma\text{Softmax-Q}_\beta(s').$$

The Boltzmann softmax distribution emerges as a prevalent method, extensively employed for action selection [Cesa-Bianchi et al. 2017; Sutton and Barto 1998] and policy optimization [Haarnoja et al. 2018; Song et al. 2019]. The main benefit of using $\mathcal{B}_{\max}$ is that we could use importance sampling to better approximate the groundtruth Bellman operator (equation 3) and reduce variance using more importance samples. Specifically, we adopt a Gaussian sampling distribution $\mathcal{N}_\phi \triangleq \mathcal{N}(\pi(s;\phi),\sigma)$ with probability density function $p^{\mathcal{N}}_\phi$ and approximate the softmax operator as follows:

$$\text{Softmax-Q}_\beta(s) \approx \frac{\mathbb{E}\left[\frac{\exp(\beta Q_{\min}(s,a))Q_{\min}(s,a)}{p^{\mathcal{N}}_\phi(a)}\middle| a\sim\mathcal{N}_\phi\right]}{\mathbb{E}\left[\frac{\exp(\beta Q_{\min}(s,a))}{p^{\mathcal{N}}_\phi(a)}\middle| a\sim\mathcal{N}_\phi\right]},$$

Note that we have also incorporated the clipped double-Q estimator in the softmax operator to mitigate over-estimation (i.e., SD3 proposed by Pan et al. [2020]). For variance reduction, drawing more samples can incur additional cost in policy and value inferences, but these additional costs are neglectable compared with the cost of data collection by fluid simulation.

Lower-bounded Bellman Operator. The estimator $\mathcal{B}_{\max}$ can mitigate over-estimation but instead introduces under-estimation due to the min-operator in $Q_{\min}$. We propose a novel operator to further mitigate under-estimation as inspired by the N-step unrolling method [Hessel et al. 2018]. Let us consider unrolling the partial trajectory once more to yield a transition tuple $\langle s,a,s',a',s''\rangle$, we could then delay the min-operator to the next timestep by defining:

$$\mathcal{B}^2_{\max}(s,a,s',a',s'') = r(s,a) + \gamma r(s',a') + \gamma^2\text{Softmax-Q}_\beta(s'').$$

Now since our goal is to mitigate under-estimation, we take the maximum over the two estimation to derive our final two-step Bellman operator as follows:

$$\mathcal{B}_{\text{2-step}}(s,a,s',a',s'') = \max\left[\mathcal{B}_{\max}(s,a,s'), \mathcal{B}^2_{\max}(s,a,s',a',s'')\right],$$

where we take the maximum over the two estimators to mitigate under-estimation. Note that plugging our two-step operator $\mathcal{B}_{\text{2-step}}$ into equation 2 would require doubling the cost of importance sampling for the softmax operator. More generally, we could unroll the trajectory over N steps and blend the solution by the maximum operator. However, through our extensive experiments in figure 7, we find that two-step unrolling strikes the best balance between performance and cost, which already leads to satisfactory results and further unrolling does not significantly improve results. Our method further mitigates the underestimation bias compared to SD3, resulting in improved convergence speed and final performance, as demonstrated in our 2D scoop benchmark shown in figure 5 and 6.

Exploration Noise. Unlike stochastic policies that explore through action sampling, our deterministic off-policy methods must rely on external action noise for exploration. The standard approaches employ either Gaussian white noise or Ornstein-Uhlenbeck (OU) red noise [Uhlenbeck and Ornstein 1930] to compensate for this inherent exploration limitation. While white noise's time-independent nature leads to inefficient exploration, red noise's temporal correlation improves exploration efficiency [Lillicrap et al. 2016]. However, its unbounded variance growth may violate action constraints. We instead adopt pink noise [Eberhard et al. 2023] as our default exploration strategy, balancing between white and red noise properties.

Fluid Effective Domain Guidance (FEDG). Achieving effective and stable control in fluid-solid coupling control tasks, particularly in multi-task and multi-goal settings, is a significant challenge due to the sparse nature of reward signals. Especially during the initial exploratory phase, this issue hampers the reinforcement learning agent's ability to secure a substantial fraction of positive rewards within complex fluid-solid coupling settings, thereby leading to either slow convergence or, in some instances, non-convergence. To mitigate this issue, the hindsight experience replay (HER) [Andrychowicz et al. 2017] can generate a sufficient number of positive samples by employing goal relabeling strategies. However, not all trajectories that result in negative rewards meet the relabeling criteria, particularly in scenarios where even suboptimal goals are unattainable. Instead, Luo et al. [2023] employ the SGES strategy to assist the robotic arm in quickly reaching different target objects, which uses a low-level policy to guide the arm's end-effector to the point where the object is located . Inspired by their work and integrating it with the fluid-solid coupling control scenarios, we propose the FEDG algorithm that extends the concept of points into complex three-dimensional spatial and temporal domains.

Algorithm 1: Data Collection using FEDG

if $r_\Omega(s) = 0$ **then**
  $a = \pi(s;\phi)$ with guiding probability and $a = \pi_\Omega(s;\phi)$ o.w.
else
  $a = \pi(s;\phi)$ o.w.
end
Add pink noise to action a
Execute action a to yield s' and observe $r(s,a)$ and $r_\Omega(s)$
$\mathcal{D} \leftarrow \mathcal{D} \cup \{\langle s, a, r, s'\rangle\}$
if $r_\Omega(s) = 0$ **then**
  $\mathcal{D}_\Omega \leftarrow \mathcal{D}_\Omega \cup \{\langle s, a, r_\Omega, s'\rangle\}$
end

Specifically, we suppose the user could define a sub-task that takes the form of a sub-goal region Ω, such that reaching the region could help in achieving the final goal. Therefore, we could define another reward signal $r_\Omega(s) = \mathbb{I}_\Omega(s)$, where $\mathbb{I}_\Omega(s)$ is the indicator function that equals to 1 iff the sub-goal is reached and $\mathbb{I}_\Omega(s)$ is less sparse by the design of Ω. FEDG works by training the policy to reach the sub-goal region Ω first and then achieve the final objective. Specifically, we train two policies π and π_Ω, which are optimal policies for reward signals r and r_Ω, respectively. We further design the two policies to use a shared architecture so that training the optimal policy π_Ω for the less sparse reward r_Ω could provide useful guidance for training π. Specifically, we introduce a policy π_{FEDG} with an augmented state space having an additional bit indicating whether the policy is π or π_Ω. In other words, we define $\pi(s;\phi) = \pi_{\text{FEDG}}(s,0;\phi)$ and $\pi_\Omega(s;\phi) = \pi_{\text{FEDG}}(s,1;\phi)$. During training, we evaluate $r_\Omega(s)$ to see if the sub-goal has been reached. If $r_\Omega(s) = 1$, then we use the optimal policy for the original task, setting action to be $a = \pi(s;\phi)$. Otherwise, we choose between the action proposed by $\pi(s;\phi)$ and $\pi_\Omega(s;\phi)$ with a predefined probability, denoted as guiding probability. Correspondingly, we store two replay buffers $\mathcal{D}$ and $\mathcal{D}_\Omega$ for training π and π_Ω, respectively. Suppose $r_\Omega(s) = 1$, we only populate $\mathcal{D}$ with new transition tuples using r as the reward signal. Otherwise, we populate both $\mathcal{D}$ and $\mathcal{D}_\Omega$ with transition tuples using r and r_Ω as the reward signals, respectively. The data collection step for FEDG with pink noise is summarized in algorithm 1.

5 Evaluation

In this section, we design a series of 2D and 3D benchmark tasks to evaluate our method. When comparing the performance of optimized controllers, we always set a same number of state transition tuples that can be generated for training the controller.

Table 1. A comparison of policy performance evaluated over 2000 episodes for each variant of the squeeze task. Our method outperforms all others on every metric in all control tasks.

Method	Double Walls		Single Wall		Target Balls				
	#Balls	Reward	#Balls	Reward	#Good	#Bad	G/B	Reward	#Steps
SD3	4.48	802.99	1.23	504.63	2.13	0.29	7.26	365.89	247.59
TD3	4.38	783.74	1.09	486.04	1.97	0.43	4.55	331.59	215.54
SAC	4.42	754.39	1.37	479.55	2.13	3.57	5.98	359.83	232.34
Ours	**4.51**	**821.22**	**1.76**	**622.48**	**2.42**	**0.26**	**9.16**	**426.13**	**257.56**

5.1 Squeeze Benchmark (2D)

As illustrated in figure 3, our first benchmark involves a 2D tank with movable walls at the boundaries, enabling solid balls within the fluid to enter a net located at the center bottom. Balls can only enter from directly above the net and can only enter but not exit. The net does not interact with the fluid but has collision detection against the ball. We consider three different variants of this benchmark.

Squeeze with Double Walls. In this task, walls are present on both the left and right sides, capable of horizontal movement within a specified range, with the goal of maximizing the number of balls that enter the net. The action a comprises of two variables, each taking values in the range $[-1, 1]$ representing the acceleration to the left and right walls, respectively. The state components d and q consists of the position and velocity of the walls and balls, respectively, and features of the fluid velocity field. The reward signal is the number of balls in the net and the maximal episode length is 300.

Squeeze with Single Wall. To demonstrate that our method outperforms for complex tasks, we increase the difficulty and design two additional tasks. In this single-wall case, we remove the right wall, reduce the net size, and set the maximal episode length to 500. All other settings remain the same.

Squeeze Target Balls. In this case, based on the double-wall task, 2 balls further are designated as bad balls, and the remaining 3 as good balls. The objective is to maximize the number of good balls that enter the net while preventing any bad balls from entering. The state s is augmented with a five-dimensional binary vector c, where $c_i = 1$ iff the i-th ball is a bad ball. The reward r is defined as the number of good balls minus the number of bad balls in the net. All other settings stay the same as in the double-wall task.

For evaluation, we compare our method with prominent RL methods including TD3 [Fujimoto et al. 2018], SAC [Haarnoja et al. 2018],

Fig. 3. (Left)We illustrate snapshots of trajectories for variants of 2D squeeze benchmarks. On the top row, we show frames of a successful trajectory for squeeze with double walls, where the goal is to control the double walls on the left and right to maximize the number of solid balls into the net. In the middle row, we show the squeeze with only a single left wall. On the bottom row, we show squeeze for the set of three target good balls, while excluding the two bad balls. (Right) The converge curves for RL training on variants of the squeeze benchmark: squeeze with double walls (top), squeeze with single wall (middle), and squeeze target balls (bottom). The results show that our method consistently performs no worse than the others. Moreover, on more challenging tasks, including the squeezing with single wall and squeezing target balls, our method achieves faster convergence and better final performance.

and SD3 [Pan et al. 2020], where SAC is the RL method used in prior state-of-the-art [Ren et al. 2022]. For fairness, we are not using FEDG in all RL method. Each policy is trained by collecting five million transition tuples and is evaluated over 2000 episodes. The convergence histories are summarized in figure 3. For squeezing with double walls, our method achieves an average reward of 821.22 and an average of 4.51 balls into the net, outperforming all other methods, although the improvement is not major. For the more challenging task of squeezing with single wall, our average number of balls is 1.76, which is 28% better than the second to best (SAC). The reward is 622.48, which is 23% higher than the second to best (TD3). For the hardest task of squeezing the target ball, the G/B index of our method is 69% better than SAC and 26% better than SD3 as shown in table 1. Furthermore, the achieved reward is 426.13, representing an increase of at least 16% compared to others.

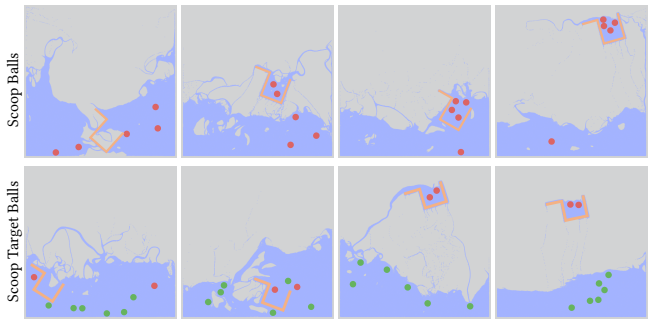

Fig. 4. The illustration of the scoop benchmark. (a) Scoop balls, the goal is to control the spoon to scoop as many balls as possible from the fluid. (b) Scoop target balls, is to scoop all the red balls and keeps out the green balls.

5.2 Scoop Benchmark (2D)

Scoop benchmark, originally proposed by Ren et al. [2022], involves using spoons to scoop balls from a tank. Our action a involves the directional and angular accelerations for the spoon. Similar to the case with squeeze, we consider two variants of the scoop task.

Scoop Balls. In this task, the tank contains five solid balls, and the goal is to control the spoon to scoop as many balls as possible. The state component q consists of the position and velocity of each ball, and the component d consists of the (angular and linear) position and velocity of the spoon. The reward signal is defined as follows:

$$\begin{cases} r(s,a) = \omega_1 f(x_s^\star - x_s) + \mathbb{I}(\#_{\text{in}} \geq 2)\left[\omega_2 f(\dot{x}_s) + \omega_3 f(\theta_s)\right] \\ f(\bullet) = \#_{\text{in}}^2 \exp(-\|\bullet\|^2) \end{cases}, \quad (5)$$

where x_s and $x_s^\star$ are the position and target position for the spoon, θ_s is the angular position of the spoon, and $\#_{\text{in}}$ is the number of balls in the spoon. The goal is to get as many balls as possible in the spoon and get the spoon in a rest, upright position at the end. ω_1, ω_2, ω_3 are coefficients set to 2, 0.25, and 0.25, respectively. Each episode terminates upon reaching the maximum number of 150 timesteps.

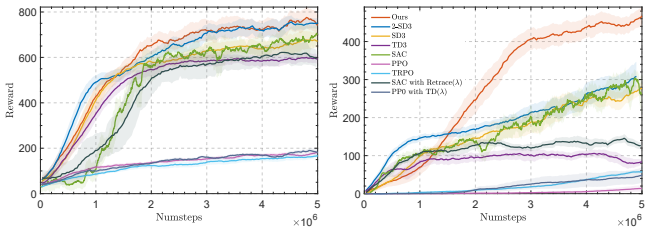

Fig. 5. The convergence histories on variants of the scoop task: scoop balls (left) and scoop target balls (right). Our method achieves a higher final reward compared to other methods, particularly on the more challenging task of scooping target balls. The two subfigures have a shared legend.

Scoop Target Balls. In this case, we specify 2 good balls and 6 bad balls, where the balls are indicated again using an augmented binary vector c. Our goal is to scoop as many good balls as possible, while leaving the bad balls out. To achieve this, we redefine the function as $f(\bullet) = (\omega^+ \#^+_{\text{in}} - \omega^- \#^-_{\text{in}}) \exp(-\|\bullet\|^2)$, where $\#^+_{\text{in}}$ and $\#^-_{\text{in}}$ are the number of good and bad balls in the spoon and the coefficients ω^+, ω^- are set to 1 and 0.4. All other settings are the same.

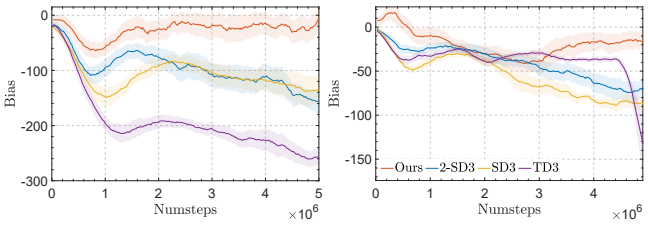

Fig. 6. The biases for scoop balls (left) and scoop target balls (right) are the difference between the Q value and discounted Monte Carlo return. Less than 0 means underestimation.

To provide a comprehensive comparison, we evaluate against TRPO [Schulman et al. 2015], PPO [Schulman et al. 2017], and methods leveraging multi-step information—including a 2-step returns variant of SD3 (2-SD3), PPO with TD(λ) [van Seijen and Sutton 2014], and SAC with Retrace(λ) [Munos et al. 2016]—alongside baseline methods from the squeeze benchmark. Again, we exclude FEDG for fairness. We first consider the easier task of scoop balls. As illustrated in figure 5, after collecting approximately 2.5 million transition tuples, our method attains the same controller performance as one trained using SD3 and SAC. As shown in table 2, our method achieves the best performance, with an average of 0.53 more balls collected than SAC, representing a 26% increase. Additionally, our average score is 73 points higher than SAC, exceeding 700. For the more challenging task of scooping target balls, our method achieves the same performance as SAC and SD3 after collecting 2.5 million transition tuples. Our cumulative reward at final convergence reaches around 470, which is more than 100 points higher than that of SD3 and SAC. From table 2, our success rate reaches 91.8%, while the other methods only reach 70%. The results demonstrate that our method outperforms both state-of-the-art on-policy (PPO) and off-policy (SAC, TD3, SD3) methods. Compared with other multi-step methods (i.e., 2-SD3, PPO with TD(λ), SAC with Retrace(λ)), our method delivers more stable performance and faster convergence in complex tasks. As evidenced in figure 6, our method significantly reduces estimation bias, directly contributing to its superior performance.

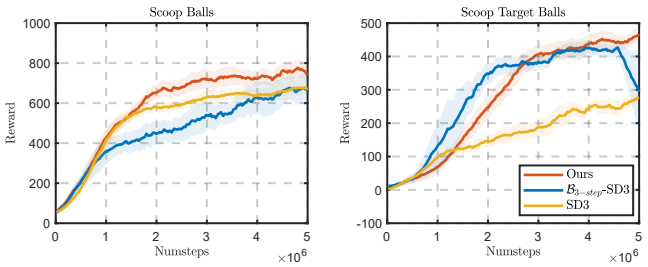

Fig. 7. We extend our Bellman operator to expand 3-steps, which is incorporated into SD3 (denoted as $\mathcal{B}_{3\text{-step}}$-SD3) and evaluate its performance on both scoop benchmark tasks. The performances of $\mathcal{B}_{2\text{-step}}$ and $\mathcal{B}_{3\text{-step}}$ are comparable.

To better compare sample efficiency between off-policy and on-policy methods, we execute the widely-used on-policy algorithm PPO on the scooping balls task for 50 million interaction steps and compare the number of interactions required for our method to achieve the same level of performance. As shown in figure 8, our method achieves comparable performance with only 1 million interactions, which demonstrates that PPO is stable but slow, and off-policy algorithms are more suitable for fluid-solid coupling scenarios where simulation costs are high.

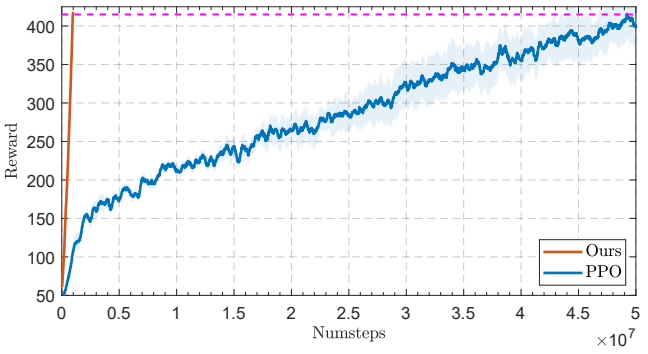

Fig. 8. Comparison of sample efficiency between PPO and our method on task of scooping balls. After training the on-policy algorithm PPO for 50 million interaction steps, we measured the number of interactions required by our off-policy method to achieve comparable performance. Our approach reaches similar performance with only approximately 1 million interactions—50 times more sample-efficient than PPO.

Additionally, as shown in figure 9, when generalizing the policy trained on a 128 grid resolution simulation to higher resolutions, our method experiences a notably smaller performance decrement compared to SAC. Notably, our method sustains a success rate of over 50% at a grid resolution above 256, while the success rate of SAC

Table 2. The performance of optimized controller on variants of the scoop task, evaluated over 2,000 episodes for each task.

Tasks	Metric	Method						
		SD3	2-SD3	TD3	PPO	TRPO	SAC	Ours
Scoop Balls	#Balls	2.02	2.34	2.00	0.89	0.84	2.06	**2.59**
	Reward	648.14	711.86	607.35	182.89	171.13	672.64	**746.15**
Scoop Target	Suc. Rate	70.1	69.8	0.1	0.1	0.1	70.0	**91.8**
	Reward	314.14	310.64	114.13	30.31	60.19	336.74	**475.10**

consistently declines to around 15%. This suggests that our method captures more salient information, thereby exhibiting enhanced generalization and robustness.

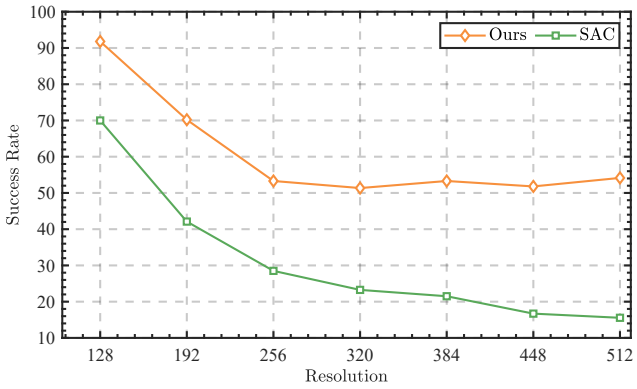

Fig. 9. We increase the grid resolution of MPM simulator and evaluate on the task of scooping target balls, using a policy trained on a 128×128 grid resolution. The success rate is tested over 2000 episodes, where our method consistently experiences less performance drop.

5.3 Balance Benchmark (3D)

The balance task is a 3D extension of the 2D version proposed by Ma et al. [2018], with the goal of spraying a solid ball with fluid spouts to keep its position in the air.

Single Ball Balance. To balance a single ball, our reward signal is:

$$r(s,a) = \omega_1 \exp(-\|p_b^\star - p_b\|^2) + \omega_2 \exp(-\|\dot{p}_b\|^2) - \omega_3 \mathbb{I}_{\text{bound}}(p_b), \quad (6)$$

where p_b and $\dot{p}_b$ are the position and velocity of the solid ball, while $\mathbb{I}_{\text{bound}}$ indicates whether the solid ball hits the boundary of the domain. Essentially, our first term requires the ball to reach the target position $p_b^\star$ and the second term penalizes its velocity, while the last term prevents the ball from hitting the domain boundary. To accelerate computation, we adopt FEDG by noting that to properly control the ball, the fluid spout should first move approximately under the ball. Therefore, we design the sub-goal as:

$$\Omega = \{\langle p_b, p_s\rangle | d_{xy}(p_b, p_s) \le \bar{d}_{xy}\},$$

where p_s is the position of the fluid spout, d_{xy} measures the distance between two objects on the XY-plane, and finally $\bar{d}_{xy}$ is a user-defined upper bound on the distance. As illustrated in figure 10, our controller trained with FEDG exhibits a more intuitive controller strategy by moving the spout directly under the ball, while the one without FEDG keeps the spout unnecessarily tilted.

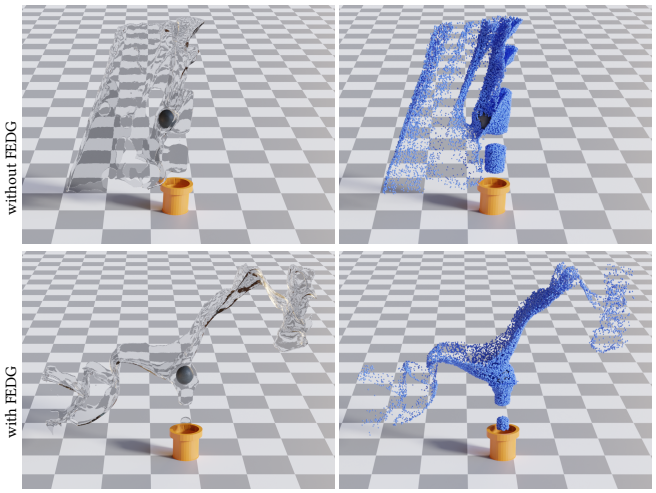

Fig. 10. Snapshots of the single ball balance task, trained by controllers with and without FEDG. With FEDG, our controller exhibits a more natural strategy, by directly moving the spout below the ball.

Table 3. Keep times (interaction steps) when the policy trained on the ball is directly transferred to balance other shapes, evaluated over 2,000 episodes.

Shape	Cube	Cross	Octahedron
Keep Times	1913	1340	3227

To demonstrate the generalizability of our method, we also conduct training and testing on various geometric shapes (e.g., cube, cross, octahedron) for balance tasks, with results visualized in figure 11. Furthermore, to evaluate the robustness of our control policy, we directly apply the strategy trained on the ball to balance other shapes, and the stabilization durations are in table 3. The results confirm the strong generalization capability and stability of our method across diverse configurations.

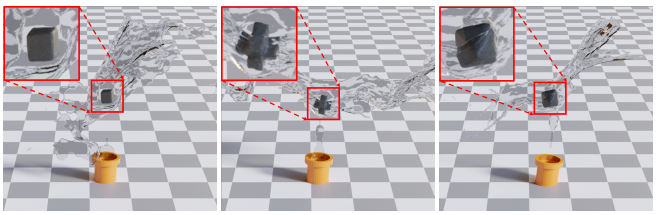

Fig. 11. Snapshots of balancing non-spherical shapes: cube (left), cross (middle), and octahedron (right). A close-up of each shape is shown at top left.

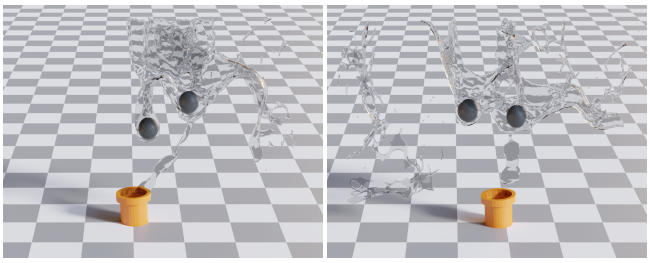

Fig. 12. Snapshots of the double ball balance task, where our controller trained with FEDG moves the spout below the ball with a lower altitude to prevent both ball from falling too far.

Double Ball Balance. For a more challenging task, we use a single spout to control two balls by summing over the reward in equation 6 for each ball. We could also use FEDG in this case by noting that the fluid spout should move under one of the two balls, and it is better to move to the ball with a lower altitude, which requires more immediate upward forces. Therefore, we define our sub-goal as:

$$\Omega = \{\langle p_b^{1,2}, p_s\rangle | d_{xy}(p_b^{j^\star}, p_s) \le \bar{d}_{xy}, j^\star = \text{argmin}_{j=1,2}[p_b^j]_z\},$$

where we use a superscript to index the ball and $j^\star$ is the index with a lower altitude. Our trained controller illustrated in figure 12 operates in exactly the expected manner. We find that our method converges much slower without FEDG. Further, without FEDG, our trained policy can only keep the ball balanced over 300 timesteps, whereas with FEDG, the ball can be balanced over 2000 timesteps.

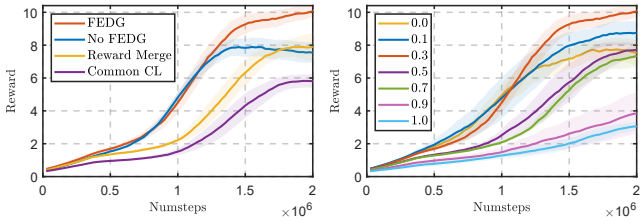

Fig. 13. The convergence curves for the double ball balance task. We compare our FEDG with common curriculum learning (CL) and the trivial method of summing up the low and high rewards (Reward Merge) (left). We also conducted an ablation study on the guide probability for FEDG (right).

To further illustrate the difference between FEDG and common curriculum learning (CL), where the low-level reward is used for the first 25% of timesteps and the high-level reward for the rest, as well as the impact of non-hierarchical reward settings, we conduct comparisons on the double ball balance task and perform an ablation study on the guiding probability. As shown in the figure 13, our method not only converges faster but also achieves a higher reward. The ablation study suggests that an optimal guidance probability of 0.3 (our default setting) effectively balances exploration guidance with minimal disruption to high-level policy learning.

5.4 Transport Benchmark (3D)

In these tasks, we employ a single spout to transport the ball to different target locations at fixed time intervals, which includes two variants: transport only in the X-axis and transport in all three axes. We use a right-handed system with the X-axis pointing to the right, the Y-axis pointing up and the Z-axis pointing out.

Transport in X-axis. In this task, we ensure relative stability along the y-axis and z-axis, while achieving transportation to the target x-axis position. When p_b reaches $p_b^\star$ within t timesteps, we update the target position at random. Otherwise, we simply terminate the episode. The reward function is designed as follows:

$$r(s,a) = \omega_1 \exp(-\|p_b^\star - p_b\|^2) + \omega_2 \exp(-\|\dot{p}_b\|^2) - \omega_3 \mathbb{I}(|[p_b^\star]_x - [p_b]_x| > \bar{d}_x),$$

where the first two terms play the same role as in equation 6, while the third term penalizes the state if the target position along the X-axis is not reached. For this task, we could also use FEDG by setting $\Omega \triangleq \left\{p_b \middle| |[p_b]_x - (\lfloor [p_b^\star]_x/\bar{d}_x \rfloor + 1/2)\bar{d}_x| < \bar{d}_x/2\right\}$, which essentially divides the domain into blocks of size $\bar{d}_x$ and requires the ball to be within the block that contains the target position.

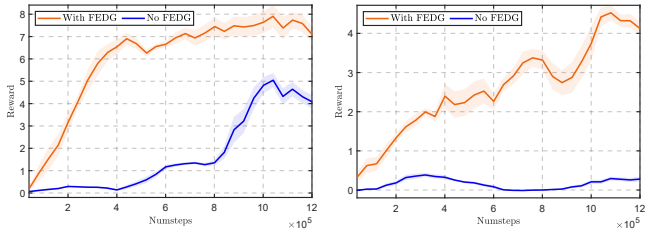

Fig. 14. The convergence curves for the two variants of the transport task: Transport in X-axis (left) and transport in 3D space (right). Comparing scenarios with and without the FEDG algorithm, we demonstrate that efficient exploration with FEDG significantly improves convergence performance.

Transport in 3D Space. In this task, we aim to transport the ball along the X-, Y- and Z-axes simultaneously. We partition the whole space into $3 \times 2 \times 2$ cubes along the three axes, randomly select one of the cubes, and transport the ball to the center of the cube. The reward function takes the following even simpler form:

$$r(s,a) = \omega_1 \exp(-\|p_b^\star - p_b\|^2) + \omega_2 \exp(-\|\dot{p}_b\|^2), \tag{7}$$

and we similarly employ FEDG by setting:

$$\Omega \triangleq \left\{ p_b \middle| \begin{array}{l} |[p_b]_x - (\lfloor [p_b^\star]_x/\bar{d}_x \rfloor + 1/2)\bar{d}_x| < \bar{d}_x/2 \\ |[p_b]_z - (\lfloor [p_b^\star]_z/\bar{d}_z \rfloor + 1/2)\bar{d}_z| < \bar{d}_z/2 \end{array} \right\},$$

which essentially requires the ball to be within the target cube of size $\langle \bar{d}_x, \bar{d}_z \rangle$ in the horizontal XZ-plane.

The snapshots of our trained controller are given in figure 16 and the convergence history with and without FEDG is presented in figure 14. For such complex tasks, RL agents without FEDG either converge very slowly for the easier case of transport in X-axis, or fail to converge for the harder case of transport in 3D space, while FEDG significantly boosts the performance.

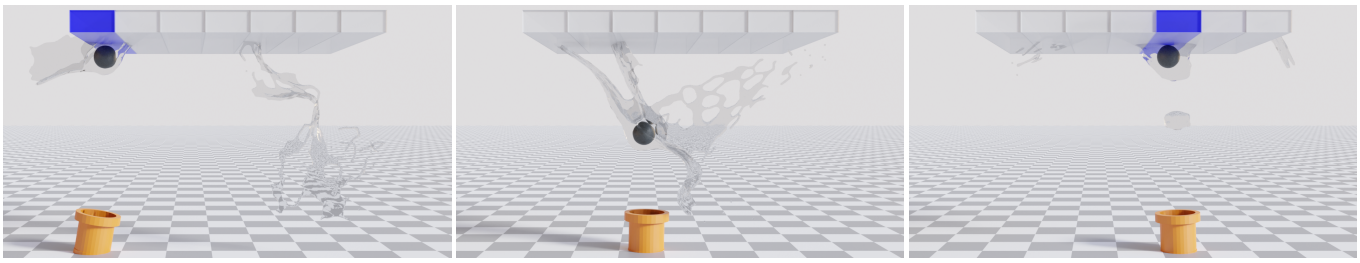

Fig. 15. Snapshots of single-ball music player, which play music at a relatively fixed tempo. After hitting a key, our robust spout controller will first catch the ball and then move to the next target directly under the key and wait for the time to hit it.

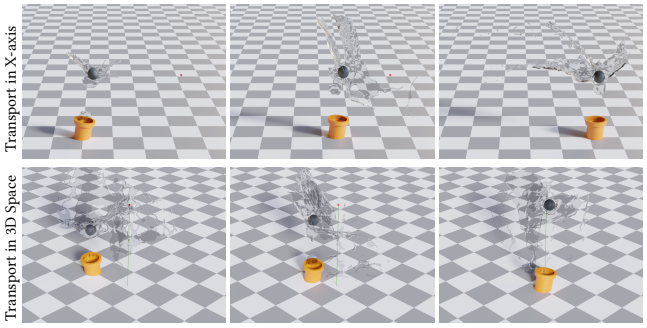

Fig. 16. Our method successfully trains robust controllers for the transport tasks with FEDG. We refer readers to the video for the full demo.

5.5 Music Benchmark (3D)

The final benchmark extends the 3D multi-solid music player benchmark in Ren et al. [2022], using fluid spouts to control balls that hit keys at the top to play music. In this benchmark, we control each ball with a separate spout to achieve more complex multi-goal tasks.

5.5.1 Single Solid Music Palyer. In the single solid case, we employ a single spout and a single ball within a rectangular simulation area to play the music. The upper section of the area is subdivided into seven equal segments along the x-axis, with each segment containing a key corresponding to the seven musical notes. To play a music script, the ball must hit the target key indexed as $k^\star$ at a specified time $t^\star$. To this end, we design the reward as:

$$r(s,a) = \mathbb{I}_{\text{hit}}(s)(\mu_1 - \mu_2|t - t^\star| - \mu_3|k - k^\star|) - \mu_4(1 - \mathbb{I}_{\text{hit}}(s)),$$

where $\mathbb{I}_{\text{hit}}$ is an indicator of hitting the key indexed by k at timestep t. However, if we directly train using the sparse reward, the algorithm fails to converge, which is again mitigated by FEDG. To define the sub-goal, we aim to first move the ball under the target key $k^\star$ with a specified time range denoted as Δt, so we set $\Omega = \{p_b \,|\, p_b \text{ under key } k^\star \text{ and } |t - t^\star| < \Delta t\}$. We further notice that in order to move the ball under the target key, we can provide a better-conditioned guidance by using a denser reward in the same form as equation 7. Therefore, instead of using $r_\Omega = \mathbb{I}_\Omega$, we set:

$$r(s,a) = \omega_1 \exp(-\|p_b^\star - p_b\|^2) + \omega_2 \exp(-\|\dot{p}_b\|^2) + \omega_3 \mathbb{I}_\Omega(s),$$

where we set $p_b^\star$ to be the point below the target key. As illustrated in figure 15, our policy demonstrates the ability to play music with a relatively fixed tempo through efficient exploration with FEDG. Moreover, in tests involving randomly triggered notes, our method achieves a success rate of 85%.

Double Solid Music Player. In the more challenging variant, we perform a variable-tempo music-playing task. We use two spouts and two balls to play more complex music scripts. The reward and FEDG setting stays the same as the single-solid case, but we use two separate rewards for each ball and sum them up. The efficacy of our method is demonstrated by its ability to facilitate multi-tempo musical execution. During the random testing phase, the hit precision is averaged at 75%.

6 CONCLUSION

We propose several enhancements to the actor-critic RL framework to solve a row of challenging solid-fluid coupled control tasks. Our main contributions involve a more accurate Q-value estimator, which uses a two-step trajectory unrolling to mitigate the underestimation. We further proposes a policy guiding approach that stochastically blend a high- and low-level policy for more efficient exploration under sparse rewards. Our results highlight the improved convergence and controller performance under a row of complex 2D and 3D benchmarks.

Our main limitation lies in the inaccurate timing control of the learned policy and the still-high overall training cost. To this partly due to the inherent high cost of the underlying simulator. In the future, we plan to incorporate distributed training, enabling faster data collection and training on more complex tasks. Recent works on differentiable simulators such as [Li et al. 2023] could also be incorporated to enable model-based RL for better sampling efficacy. Finally, our novel RL algorithm is general-purpose and we plan to evaluate it on more general benchmarks.

Acknowledgments

This work is supported by the Natural Science Foundation of China (62272245, 62441218) and the Fundamental Research Funds for the Central Universities (Nankai University, 63253233). The computation resources are provided by the Supercomputing Center of Nankai University (NKSC).

REFERENCES

Simon Alexanderson, Rajmund Nagy, Jonas Beskow, and Gustav Eje Henter. 2023. Listen, Denoise, Action! Audio-Driven Motion Synthesis with Diffusion Models. *ACM Trans. Graph.* 42, 4, Article 44 (July 2023), 20 pages. doi:10.1145/3592458

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, California, USA. https://proceedings.neurips.cc/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf

Markus Becker and Matthias Teschner. 2007. Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '07)*. Eurographics Association, Goslar, DEU, 209–217.

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Trans. Graph.* 38, 6, Article 206 (Nov. 2019), 11 pages. doi:10.1145/3355089.3356536

Nicolò Cesa-Bianchi, Claudio Gentile, Gábor Lugosi, and Gergely Neu. 2017. Boltzmann exploration done right. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6287–6296.

Kang Chen, Zhipeng Tan, Jin Lei, Song-Hai Zhang, Yuan-Chen Guo, Weidong Zhang, and Shi-Min Hu. 2021. ChoreoMaster: choreography-oriented music-driven dance synthesis. *ACM Trans. Graph.* 40, 4, Article 145 (July 2021), 13 pages. doi:10.1145/3450626.3459932

Rui Chen, Mingyi Shi, Shaoli Huang, Ping Tan, Taku Komura, and Xuelin Chen. 2024b. Taming Diffusion Probabilistic Models for Character Control. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 67, 10 pages. doi:10.1145/3641519.3657440

Yixin Chen, David Levin, and Timothy Langlois. 2024a. Fluid Control with Laplacian Eigenfunctions. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 44, 11 pages. doi:10.1145/3641519.3657468

Kyungmin Cho, Chaelin Kim, Jungjin Park, Joonkyu Park, and Junyong Noh. 2021. Motion recommendation for online character control. *ACM Trans. Graph.* 40, 6, Article 196 (Dec. 2021), 16 pages. doi:10.1145/3478513.3480512

Mengyu Chu, Nils Thuerey, Hans-Peter Seidel, Christian Theobalt, and Rhaleb Zayer. 2021. Learning meaningful controls for fluids. *ACM Trans. Graph.* 40, 4, Article 100 (July 2021), 13 pages. doi:10.1145/3450626.3459845

Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. 2019. Better exploration with optimistic actor-critic. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Vancouver, BC, Canada.

Onno Eberhard, Jakob Hollenstein, Cristina Pinneri, and Georg Martius. 2023. Pink Noise Is All You Need: Colored Noise Exploration in Deep Reinforcement Learning. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)*. OpenReview.net, Kigali, Rwanda. https://openreview.net/forum?id=hQ9V5QN27eS

Raanan Fattal and Dani Lischinski. 2004. Target-driven smoke animation. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 441–448. doi:10.1145/1015706.1015743

Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholm, Sweden, 1587–1596. https://proceedings.mlr.press/v80/fujimoto18a.html

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholm, Sweden, 1861–1870. https://proceedings.mlr.press/v80/haarnoja18b.html

Hado van Hasselt. 2010. Double Q-learning. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2* (Vancouver, British Columbia, Canada) *(NIPS'10)*. Curran Associates Inc., Red Hook, NY, USA, 2613–2621.

Hado van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double Q-Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (Phoenix, Arizona) *(AAAI'16)*. AAAI Press, Phoenix, Arizona USA, 2094–2100.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: combining improvements in deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence* (New Orleans, Louisiana, USA) *(AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, New Orleans, Louisiana, USA, Article 393, 8 pages.

Philipp Holl, Vladlen Koltun, Kiwon Um, and Nils Thuerey. 2020a. phiflow: A differentiable pde solving framework for deep learning via physical simulations. In *NeurIPS workshop*, Vol. 2. Curran Associates Inc., DiffCVGP workshop.

Philipp Holl, Nils Thuerey, and Vladlen Koltun. 2020b. Learning to Control PDEs with Differentiable Physics. In *International Conference on Learning Representations*. OpenReview.net, Addis Ababa, Ethiopia. https://openreview.net/forum?id=HyeSin4FPB

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 2018. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.* 37, 4, Article 150 (July 2018), 14 pages. doi:10.1145/3197517.3201293

Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. 2021. SUNRISE: A Simple Unified Framework for Ensemble Learning in Deep Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, Virtual Event, 6131–6141. https://proceedings.mlr.press/v139/lee21g.html

Yifei Li, Yuchen Sun, Pingchuan Ma, Eftychios Sifakis, Tao Du, Bo Zhu, and Wojciech Matusik. 2024. NeuralFluid: Nueral Fluidic System Design and Control with Differentiable Simulation. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., Vancouver, Canada, 84944–84967. https://proceedings.neurips.cc/paper_files/paper/2024/file/9a379c1b05793d1c42dc832269834515-Paper-Conference.pdf

Zhehao Li, Qingyu Xu, Xiaohan Ye, Bo Ren, and Ligang Liu. 2023. DiffFR: Differentiable SPH-Based Fluid-Rigid Coupling for Rigid Body Control. *ACM Trans. Graph.* 42, 6, Article 179 (Dec. 2023), 17 pages. doi:10.1145/3618318

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, (ICLR 2016)*. Open Publishing, San Juan, Puerto Rico.

Yongle Luo, Yuxin Wang, Kun Dong, Qiang Zhang, Erkang Cheng, Zhiyong Sun, and Bo Song. 2023. Relay Hindsight Experience Replay: Self-guided continual reinforcement learning for sequential object manipulation tasks with sparse rewards. *Neurocomputing* 557 (2023), 126620. doi:10.1016/j.neucom.2023.126620

Pingchuan Ma, Yunsheng Tian, Zherong Pan, Bo Ren, and Dinesh Manocha. 2018. Fluid directed rigid body control using deep reinforcement learning. *ACM Trans. Graph.* 37, 4, Article 96 (July 2018), 11 pages. doi:10.1145/3197517.3201334

Antoine McNamara, Adrien Treuille, Zoran Popović, and Jos Stam. 2004. Fluid control using the adjoint method. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 449–456. doi:10.1145/1015706.1015744

Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '03)*. Eurographics Association, Goslar, DEU, 154–159.

Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. Meshless deformations based on shape matching. *ACM Trans. Graph.* 24, 3 (July 2005), 471–478. doi:10.1145/1073204.1073216

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems* 29 (2016).

Michael B. Nielsen and Robert Bridson. 2011. Guide shapes for high resolution naturalistic liquid simulation. *ACM Trans. Graph.* 30, 4, Article 83 (July 2011), 8 pages. doi:10.1145/2010324.1964978

Ling Pan, Qingpeng Cai, and Longbo Huang. 2020. Softmax deep double deterministic policy gradients. *Advances in neural information processing systems* 33 (2020), 11767–11777.

Zherong Pan and Dinesh Manocha. 2017. Efficient Solver for Spacetime Control of Smoke. *ACM Trans. Graph.* 36, 5, Article 162 (July 2017), 13 pages. doi:10.1145/3016963

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. 2019. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 5331–5340. https://proceedings.mlr.press/v97/rakelly19a.html

Brener Ramos, Felix Trost, and Nils Thuerey. 2022. Control of Two-way Coupled Fluid Systems with Differentiable Solvers. *CoRR* abs/2206.00342 (2022). doi:10.48550/ARXIV.2206.00342 arXiv:2206.00342

Bo Ren, Xiaohan Ye, Zherong Pan, and Taiyuan Zhang. 2022. Versatile Control of Fluid-directed Solid Objects Using Multi-task Reinforcement Learning. *ACM Trans. Graph.* 42, 2, Article 15 (Oct. 2022), 14 pages. doi:10.1145/3554731

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37)*, Francis Bach and David Blei (Eds.). PMLR, Lille, France, 1889–1897. https://proceedings.

mlr.press/v37/schulman15.html

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347

Lin Shi and Yizhou Yu. 2005. Controllable smoke animation with guiding objects. *ACM Trans. Graph.* 24, 1 (Jan. 2005), 140–164. doi:10.1145/1037957.1037965

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, Beijing, China, I–387–I–395.

Zhao Song, Ron Parr, and Lawrence Carin. 2019. Revisiting the Softmax Bellman Operator: New Benefits and New Perspective. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, California, USA, 5916–5925. https://proceedings.mlr.press/v97/song19c.html

R.S. Sutton and A.G. Barto. 1998. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 9, 5 (1998), 1054–1054. doi:10.1109/TNN.1998.712192

Tetsuya Takahashi, Junbang Liang, Yi-Ling Qiao, and Ming C. Lin. 2021. Differentiable Fluids with Solid Coupling for Learning and Control. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 7 (May 2021), 6138–6146. doi:10.1609/aaai.v35i7.16764

Jingwei Tang, Byungsoo Kim 0001, Vinicius C. Azevedo 0001, and Barbara Solenthaler. 2023. Physics-Informed Neural Corrector for Deformation-based Fluid Control. *Comput. Graph. Forum* 42, 2 (May 2023), 161–173. doi:10.1111/cgf.14751

Sebastian Thrun and Anton Schwartz. 1993. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale*. Lawrence Erlbaum, Pittsburgh, Pennsylvania, USA. doi:10.4324/9781315806433

N. Thürey, R. Keiser, M. Pauly, and U. Rüde. 2009. Detail-preserving fluid control. *Graph. Models* 71, 6 (Nov. 2009), 221–228. doi:10.1016/j.gmod.2008.12.007

Adrien Treuille, Antoine McNamara, Zoran Popović, and Jos Stam. 2003. Keyframe control of smoke simulations. *ACM Trans. Graph.* 22, 3 (July 2003), 716–723. doi:10.1145/882262.882337

Jonathan Tseng, Rodrigo Castellon, and C. Karen Liu. 2023. EDGE: Editable Dance Generation From Music. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Vancouver, Canada, 448–458. doi:10.1109/CVPR52729.2023.00051

G. E. Uhlenbeck and L. S. Ornstein. 1930. On the Theory of the Brownian Motion. *Phys. Rev.* 36 (Sep 1930), 823–841. Issue 5. doi:10.1103/PhysRev.36.823

Martijn van Otterlo and Marco Wiering. 2012. *Reinforcement Learning and Markov Decision Processes*. Springer Berlin Heidelberg, Berlin, Heidelberg, 3–42. doi:10.1007/978-3-642-27645-3_1

Harm van Seijen and Richard S. Sutton. 2014. True Online TD(lambda). In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, 21-26 June 2014 (JMLR Workshop and Conference Proceedings, Vol. 32)*. JMLR.org, Beijing, China,, 692–700. http://proceedings.mlr.press/v32/seijen14.html

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning* (Helsinki, Finland) *(ICML '08)*. Association for Computing Machinery, New York, NY, USA, 1096–1103. doi:10.1145/1390156.1390294

Yi-Zhe Wang, Yu-Bai Li, Nadine Aubry, Yue Hua, Zhi-Fu Zhou, Zhi-Hua Chen, and Wei-Tao Wu. 2023. Performance analysis of reinforcement learning algorithms on intelligent closed-loop control on fluid flow and convective heat transfer. *Physics of Fluids* 35, 7 (2023).

Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8 (1992), 279–292.

Zhou Xian, Bo Zhu, Zhenjia Xu, Hsiao-Yu Tung, Antonio Torralba, Katerina Fragkiadaki, and Chuang Gan. 2023. FluidLab: A Differentiable Environment for Benchmarking Complex Fluid Manipulation. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, Kigali, Rwanda. https://openreview.net/forum?id=Cp-io_BoFaE

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (July 2005), 965–972. doi:10.1145/1073204.1073298