DiffFR: Differentiable SPH-based Fluid-Rigid Coupling for Rigid Body Control

ZHEHAO LI, University of Science and Technology of China, China
QINGYU XU, University of Science and Technology of China, China
XIAOHAN YE, TMCC, College of Computer Science, Nankai University, China
BO REN*, TMCC, College of Computer Science, Nankai University, China
LIGANG LIU, University of Science and Technology of China, China

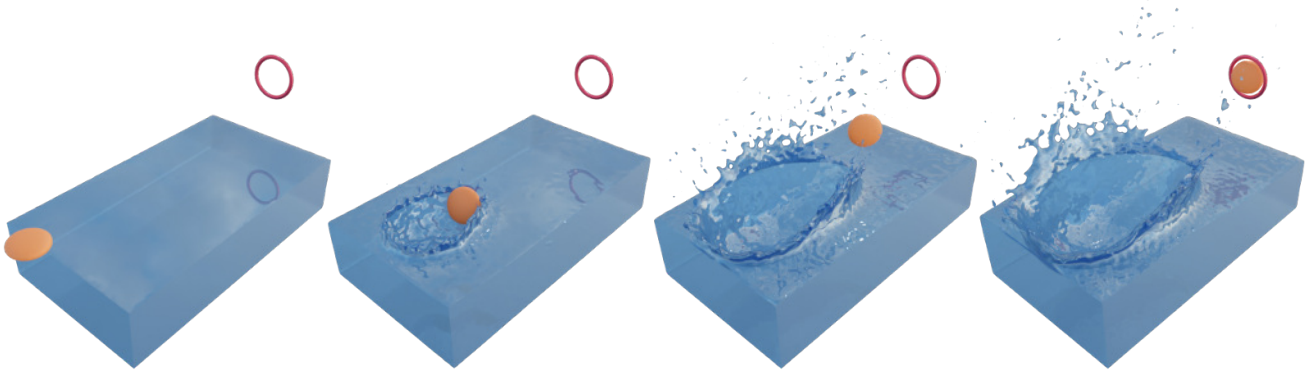


Fig. 1. Gradient-based optimization results. The stone skipping task with our differentiable SPH-based fluid-rigid coupling simulator. We optimize the initial linear and angular velocities of the stone to make it pass through the target red ring after bouncing.

Differentiable physics simulation has shown its efficacy in inverse design problems. Given the pervasiveness of the diverse interactions between fluids and solids in life, a differentiable simulator for the inverse design of the motion of rigid objects in two-way fluid-rigid coupling is also demanded. There are two main challenges to develop a differentiable two-way fluid-solid coupling simulator for rigid body control tasks: the ubiquitous, discontinuous contacts in fluid-solid interactions, and the high computational cost of gradient formulation due to the large number of degrees of freedom (DoF) of fluid dynamics. In this work, we propose a novel differentiable SPH-based two-way fluid-rigid coupling simulator to address these challenges. Our purpose is to provide a differentiable simulator for SPH which incorporates a unified representation for both fluids and solids using particles. However, naively differentiating the forward simulation of the particle system encounters gradient explosion issues. We investigate the instability in differentiating the SPH-based fluid-rigid coupling simulator and present a feasible gradient computation scheme to address its differentiability. In addition, we also propose an efficient method to compute the gradient of fluid-rigid coupling without incurring the high computational cost of differentiating the entire high-DoF fluid system. We show the efficacy, scalability, and extensibility of our method in various challenging rigid body control tasks with diverse fluid-rigid interactions and multi-rigid contacts, achieving up to an order of magnitude speedup in optimization compared to baseline methods in experiments.

CCS Concepts: • **Computing methodologies** → **Differentiable Simulation**.

Additional Key Words and Phrases: Physics-based Simulation, Differentiable Simulation, Fluid-Rigid Coupling

ACM Reference Format:
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 (December 2023), 17 pages. https://doi.org/10.1145/3618318

1 INTRODUCTION

Differentiable physics simulations have recently seen increasing attention as an efficient tool to solve inverse design problems. With its ability to directly differentiate the simulator to obtain the gradients of any differentiable performance metrics with respect to design parameters, it is proven to outperform gradient-free methods in many downstream tasks. Recent works have made progress in multi-body systems [Geilinger et al. 2020; Qiao et al. 2021; Werling et al. 2021; Xu et al. 2021a], deformable systems [Du et al. 2022; Hu et al. 2020, 2018b], cloth simulation [Li et al. 2023], fluidic devices [Du et al.

*The corresponding author

Authors' addresses: Zhehao Li, University of Science and Technology of China, China, zhehaoli@mail.ustc.edu.cn; Qingyu Xu, University of Science and Technology of China, China, liamxu123@mail.ustc.edu.cn; Xiaohan Ye, TMCC, College of Computer Science, Nankai University, China, yexiaohan@mail.nankai.edu.cn; Bo Ren, TMCC, College of Computer Science, Nankai University, China, rb@nankai.edu.cn; Ligang Liu, University of Science and Technology of China, China, lgliu@ustc.edu.cn.

0730-0301/2023/12-ART179 $15.00
https://doi.org/10.1145/3618318

2020; Li et al. 2022], one-way fluid-solid coupling [Takahashi et al. 2021], aquatic robot design [Ma et al. 2021; Nava et al. 2022], etc.

In addition to these tasks, the demands of inversely designing the motion of solid objects with fluid-solid interactions also commonly exist in daily life, such as flipping a water bottle to make it land upright on the table after spinning, or skipping a rock on a lake to achieve several bounces. In such problems, the fluid-rigid interaction significantly affects the motion of solids, and the goal is to control the final/middle state of solid objects after/during interaction with the fluid environment, with the input being the release velocity of solids. However, the differentiability of the simulation of two-way fluid-solid coupling has not yet been extensively investigated.

There are two main challenges to develop a differentiable two-way fluid-solid coupling simulator for rigid body control tasks. First, due to the inconsistency of constitutive equations of liquids and solids, as well as the ubiquitous, discontinuous contacts in fluid-solid interactions, the differentiability of fluid-solid coupling is still an open problem. On one hand, previous works [Du et al. 2020; Li et al. 2022; Ma et al. 2021; Nava et al. 2022; Takahashi et al. 2021] either focuses on one-way fluid-rigid coupling, takes strong assumptions on fluid models (e.g., Stokes flow in fluidic devices), or requires large training data for different scenes to learn fluid dynamics with neural networks, thus cannot be directly applied to our problem. On the other hand, although automatic differentiation tools can theoretically be applied to this problem, directly applying tools such as DiffTaichi [Hu et al. 2020] to a SPH-based two-way coupled fluid-rigid simulator encounters problems such as gradient explosion in practice. Second, the large number of degrees of freedom (DoF) of fluids incurs high computational costs in gradient formulation. In previous studies on differentiable simulation, the entire system being simulated typically needs to be differentiated [Hu et al. 2020; Takahashi et al. 2021], but the high DoFs and the chaotic nature of fluid dynamics distinguish it from deformable and multi-body systems, which have relatively fewer DoFs.

In this paper, we present a novel differentiable two-way fluid-rigid coupling simulator for the inverse design of control tasks of rigid bodies. To tackle the first challenge of differentiability, our purpose is to provide a differentiable simulator for Smoothed Particle Hydrodynamics (SPH) which incorporates a unified representation for fluids and solids using particles. However, naively differentiating the forward simulation of the particle system encounters gradient explosion issues. We investigate the causes of this gradient instability, which is related to the chaos in differentiating the high-DoF particle interactions and the high sensitivity of SPH kernel functions to particle distances. To handle this underlying non-smoothness of the system, we propose a feasible localized gradient computation scheme based on the idea of reducing the problem scale to address the differentiability of SPH-based fluid-rigid coupling. Here our key assumption is that in fluid-solid coupling, small perturbations to the velocities of solids generally do not significantly affect the bulk motion of the fluid environment, therefore, the problem scale can be reduced by only considering the fluid surrounding the solids in the gradient computation of fluid-rigid coupling.

To address the second challenge of efficiency, the proposed localized gradient computation scheme allows us to obtain local information about the fluid while avoiding the high computational cost associated with differentiating the entire high-DoF system, which is efficient.

Finally, we adopt our method to various challenging rigid body control tasks with versatile fluid-rigid interactions. Some of our control results have not been introduced to computer graphics yet to the best of our knowledge. We have achieved up to an order of magnitude speedup compared with gradient-free optimization methods in optimizations, demonstrating our method's effectiveness.

Our paper makes the following technical contributions:

- We investigate the instability in differentiating the particle-based fluid-rigid coupling simulator using SPH and present a feasible gradient computation scheme to address its differentiability.
- We present an efficient computational scheme to obtain the gradient of fluid-rigid coupling while avoiding the high computational cost to differentiate the entire high-DoF fluid system.
- We show the efficacy, scalability, and extensibility of our method in various fluid-rigid coupled rigid body control tasks, including multi-rigid systems and training neural network controllers.

2 RELATED WORK

2.1 Differentiable Simulation

Differentiable simulation is a relatively recent concept explored in the graphics and machine learning community. Despite the recent advances in differentiable simulators in rigid-body dynamics [Freeman et al. 2021; Geilinger et al. 2020; Qiao et al. 2021; Xu et al. 2021a], soft-body dynamics [Du et al. 2022; Hahn et al. 2019; Hu et al. 2018b; Murthy et al. 2020], cloth simulation [Li et al. 2023; Liang et al. 2019], fluid dynamics and control[Du et al. 2020; Holl et al. 2020; Li et al. 2022; McNamara et al. 2004; Schenck and Fox 2018], the work of differentiable simulation for fluid-rigid coupling is relatively rare.

To differentiate a simulator, finite difference, auto-differentiation, and manually deriving analytical derivatives, as well as their combinations are typical ways. The complex-step finite difference has recently seen increasing attention [Luo et al. 2019; Shen et al. 2021] for overcoming the drawbacks in the real number domain such as subtractive cancellation issues. Auto-differentiation tools [Hu et al. 2020, 2018b] save the labor of manual derivation of gradient, which store the information of the computational graph with a tape in the forward computation and then backpropagate the computational graph to obtain gradient.

To obtain the analytical gradient, McNamara et al. [2004] propose to use the adjoint method to efficiently compute the gradient of a fluid system for keyframe fluid animation control. Nava et al. [2022] recently propose a differentiable fluid-solid coupling method for aquatic robot design based on the immersed boundary method with fluid dynamics inferred with a neural network, which needs to be re-trained on different scenes with large training data to predict the fluid motion plausibly. Takahashi et al. [2021] propose to use the adjoint method based on the variational formulation of fluid-solid coupling [Batty et al. 2007], however, they only consider one-way coupling from solid to fluid, which limits its use-case. Li et al. [2022]

propose an anisotropic mixture model with no-slip and free-slip boundary conditions of fluid-rigid coupling for topology optimization of fluidic devices. Since our work considers the diverse two-way coupling between dynamic fluids and solid objects, these methods cannot be directly applied to our problem.

2.2 Two-way Fluid-Solid Coupling

Because the literature on the topic of general fluid-solid interactions is extensive, in this section we focus on the two-way coupling methods of fluids and rigid objects. Depending on the representation methods used for fluids, methods of two-way fluid-solid coupling can be classified into three categories: grid-based, particle-based, and hybrid-based. Various techniques for grid-based fluid-solid coupling have been developed over the years [Batty et al. 2007; Guendelman et al. 2005; Klingner et al. 2006; Narain et al. 2010; Robinson-Mosher et al. 2008; Takahashi and Batty 2020; Takahashi and Lin 2019], where solving fluid-solid coupling is usually formulated as solving a unified system to determine the correct exchange of forces between fluids and solids.

Another popular method to solve multi-material interactions is the hybrid-based Material Point Method (MPM), where collisions with different materials can be naturally handled by assigning different material properties to particles and interchanging their momentums on background grids. Due to this advantage of MPM, two-way fluid-solid coupling has been achieved in various settings [Daviet and Bertails-Descoubes 2016; Ding and Schroeder 2020; Fang et al. 2020; Han et al. 2019; Hu et al. 2018a; Klár et al. 2016].

Since the Lagrangian view is quite natural for solid objects, the Smoothed Particle Hydrodynamics (SPH) method offer another consistent framework for formulating coupling problems [Becker and Teschner 2007; Bender and Koschier 2015, 2017; Ihmsen et al. 2014; Koschier et al. 2019; Solenthaler and Pajarola 2009; Takahashi et al. 2018] where fields of physical properties of fluid are discretized with sampling points named smoothed particles. To handle the particle deficiency near the fluid-solid boundaries, Akinci et al. [2012] propose an SPH-based two-way fluid-rigid coupling method by sampling particles on the surface of solids, which will be introduced in detail since it is closely related to our method. To improve the accuracy of coupling, Band et al. [2018b] propose to use the moving least square method to extrapolate the pressure of boundary particles; Bender et al. [2019]; Koschier and Bender [2017] propose to use implicit representations for solid objects with signed distance fields. To further improve the stability of simulation and consider rigid-rigid coupling, Gissler et al. [2019] propose a strong two-way coupling scheme for SPH with rigid-rigid contact. In our work, we will not consider such type of strong two-way coupling of fluids and solids and leave it to our future work. For rigid-rigid contact, Gissler et al. [2019] also propose an SPH-based rigid contact solver based on artificial density deviations at rigid particles. Based on the continuity equation, an equation of artificial pressure field on all rigid particles in contact is solved to resolve the rigid-rigid contact by pressure gradient forces. However, the complexity of this forward simulation method makes it non-trivial to differentiate in our practice, but inspired by this idea and [Xu et al. 2021a,b], we present a penalty-based SPH-based rigid-rigid coupling method, which has a better differentiability.

2.3 Control of Rigid Bodies with Fluid-Solid Interaction

The control of rigid bodies and multi-body systems with fluid-solid interactions is an active research topic in the field of graphics and robotics. Besides the aforementioned works that derive derivatives from physical-based simulators to get gradients of the simulation process, some researchers directly adopt learning-based simulators to leverage the differentiability of neural networks to obtain gradient information. Physics-informed networks are widely used to approximate the solution of the partial differential equations of the simulated system [Nava et al. 2022; Raissi et al. 2019; Ramos et al. 2022]. Li et al. [2018]; Pfaff et al. [2020] use graph networks or their variants to build a coupling simulator. However, it is hard to guarantee the physical accuracy of the simulation predicted by neural networks unless trained with a large amount of data. In addition to gradient-based methods, gradient-free approaches are also popular for inverse design problems. Tan et al. [2011] optimize the gait of articulated swimming creatures using gradient-free searching methods. Besides, some works assume the solid objects are directed by fluids [Ma et al. 2018; Ren et al. 2023], where techniques from reinforcement learning are adopted to achieve versatile control tasks of fluid-directed solid objects.

3 PROBLEM FORMULATION

In this section, we present a general formulation of the control problem. We focus on the control of rigid bodies in a fluid-rigid coupled system, where the fluid-rigid interaction significantly affects the motion of solids.

Input & Output. In the coupled fluid-rigid system, we represent the state $\mathbf{s}_{\mathcal{R}}$ of a rigid body $\mathcal{R}$ with its linear and angular velocity v, ω, position x and the 3-dimensional orientation q, namely $\mathbf{s}_{\mathcal{R}} := (v, \omega, x, q)$, as shown in Fig. 2. The state $\mathbf{s}_{\mathcal{F}}$ of the fluid $\mathcal{F}$ is discretized with Lagrangian particles carrying physical properties. The inputs of the control problem are the initial state $\mathbf{s}^0_{\mathcal{R}}, \mathbf{s}^0_{\mathcal{F}}$, and the target rigid body state $\mathbf{s}^n_{\mathcal{R}}$ at a user-specified time step n. The expected output is the optimal elements of the initial rigid body state $\mathbf{s}^0_{\mathcal{R}}$ to make $\mathcal{R}$ reach $\mathbf{s}^n_{\mathcal{R}}$ after simulation for n time steps.

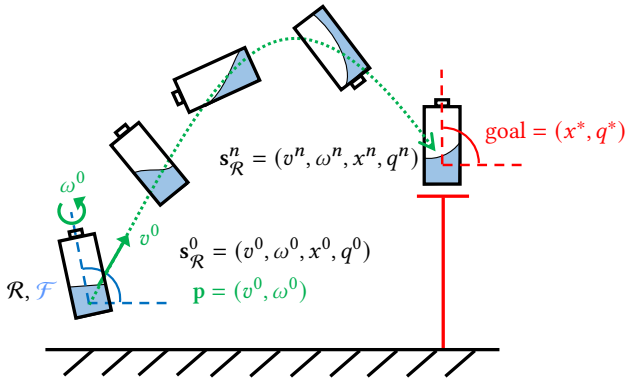


Fig. 2. Problem formulation of one of the control tasks "water bottle flip challenge" as an example. The design variables (green color) are the initial velocity $\mathbf{p} = (v^0, \omega^0) \subset \mathbf{s}^0_{\mathcal{R}}$. The goal (red color) is to land on target position x^* while reaching the target orientation q^* at a user-specified time.

Formulation. We formulate this inverse design of rigid body control task as an optimization problem. We define the goal of our task as a differentiable metric of target rigid body state $\mathbf{s}_{\mathcal{R}}^n$ at the user-specified time step n of a simulation trajectory as $E(\mathbf{s}_{\mathcal{R}}^n)$, which is to be minimized: $\min_{\mathbf{p}} E(\mathbf{s}_{\mathcal{R}}^n) = E_{\text{goal}} + \lambda E_{\text{penalty}}$, where $\mathbf{p}$ is the design variable introduced below, and E_{penalty} is the penalty term that describe the constraints with λ as the weight parameter.

Design Variables. We define the design variable $\mathbf{p}$ of the control tasks as the elements of the initial state $\mathbf{s}_{\mathcal{R}}^0$ of $\mathcal{R}$. One of the examples is the "water bottle flip challenge" task, in which the design parameters are the initial release velocity $(v^0, \omega^0) \subset \mathbf{s}_{\mathcal{R}}^0$ of the bottle $\mathcal{R}$, and the goal is to make the bottle land on target position while standing upright on its base or cap, with E_{goal} defined as the combination of linear and angular distances to the target, as Fig. 2 shows. This proposed formulation can be readily extended to control tasks in which the control signal is applied to rigid body agents at regular intervals of N time steps to achieve a continuous control.

4 FORWARD SIMULATION

We present a brief overview of the forward simulation of SPH-based fluid, fluid-rigid coupling, and rigid body dynamics. For rigid-rigid coupling, we adopt a penalty-based method to develop a unified differentiable SPH-based fluid-rigid coupled system.

4.1 SPH-based Fluid Simulation

We simulate fluids using the Navier-Stokes equations for incompressible flow in Lagrangian coordinates. We follow the operator splitting scheme and adopt the implicit DFSPH method [Bender and Koschier 2015, 2017] for its good stability and efficacy. DFSPH solves the pressure projection in an iterative manner, and the pressure p_{f_i} of fluid particle f_i is solved implicitly with the source term respectively being the divergence error and density error as:

$$p_{f_i} = \frac{1}{\Delta t}\frac{D\rho_{f_i}}{Dt} k_{f_i}^{\text{DFSPH}}, \quad p_{f_i} = \frac{1}{\Delta t^2}\left(\rho_{f_i}^* - \rho_0\right) k_{f_i}^{\text{DFSPH}}, \tag{1}$$

where $k_{f_i}^{\text{DFSPH}}$ is a before-iteration pre-computed factor, $\frac{D\rho_{f_i}}{Dt} = \sum_j m_j \left(\mathbf{v}_i - \mathbf{v}_j\right) \cdot \nabla W_{ij}$ is the density error caused by velocity advection, where $W_{ij} = W(x_i - x_j, h)$ is the kernel function with support radius h and two particle positions x_i, x_j; $\rho_{f_i}^* = \rho_{f_i} + \Delta t \frac{D\rho_{f_i}}{Dt}$ is the predicted density. The density error solver and the divergence error solver in DFSPH have a similar structure as shown in Alg. (1). In the pressure projection, the coupling force with solid boundaries needs to be considered, which is introduced in the next subsection.

4.2 SPH-based Two-way Fluid-Rigid Coupling

For fluid-solid coupling, we adopt the method of [Akinci et al. 2012], in which the surfaces of rigid bodies are sampled with boundary particles to prevent deficiency issues, and the two-way coupling is solved based on hydrodynamic forces in a momentum-conserving manner. In [Akinci et al. 2012], the density ρ_{f_i} of fluid particle f_i reads as: $\rho_{f_i} = m_{f_i} \sum_{f_j} W_{ij} + \sum_{b_j} \Psi_{b_j}(\rho_0) W_{ij}$ where $\Psi_{b_j}(\rho_0) = \rho_0 V_{b_j}$ is the contribution of a boundary particle b_j by taking the volume V_{b_j} of b_j into account, where $V_{b_j} = \frac{m_{b_j}}{m_{b_j} \sum_{b_i} W_{ji}}$

ALGORITHM 1: Pressure Projection Solver in DFSPH

```
Function PressureProjection (k^DFSPH)
    ε = density error (or divergence error)
    while (ε > thresh) do
        parallel forall fluid particle f_i do
            compute ρ_{f_i} (or Dρ_{f_i}/Dt )
        parallel forall fluid particle f_i do
            sequential forall neighbor fluid particles f_j do
                κ_i = (ρ*_i − ρ_0)/Δt^2 k_i^DFSPH (or (1/Δt Dρ_i/Dt) k_i^DFSPH)
                κ_j = (ρ*_j − ρ_0)/Δt^2 k_j^DFSPH (or (1/Δt Dρ_j/Dt) k_j^DFSPH)
                v*_i := v*_i − Δt Σ_j m_j (κ_i/ρ_i + κ_j/ρ_j) ∇W_ij
            sequential forall neighbor boundary rigid particles b_j do
                compute fluid-rigid coupling force with Eq. (2)
                update v*_i
    end
    return updated fluid particle velocities v*
end
```

with b_i denoting the neighboring boundary particles of b_j. Then the pressure force applied from a boundary particle b_j to a fluid particle f_i is derived as:

$$F_{f_i \leftarrow b_j} = -m_{f_i} \Psi_{b_j}(\rho_0) \left(\frac{p_{f_i}}{\rho_0 \rho_{f_i}}\right) \nabla W_{ij}. \tag{2}$$

Based on Newton's third law, the symmetric pressure force from fluid particle f_i to boundary particle b_j is $F_{b_j \leftarrow f_i} = -F_{f_i \leftarrow b_j}$, which naturally enforces the conservation of momentum. Finally, the net force f and torque τ of a rigid body is a double summation of all forces and torques of the rigid particles from their neighboring fluid particles, with $\tau_{b_j \leftarrow f_i} = r_{b_j} \times F_{b_j \leftarrow f_i}$ in which r_{b_j} is the displacement of rigid particle b_j to the body's mass center:

$$f = \sum_{b_j} \sum_{f_i} F_{b_j \leftarrow f_i}, \quad \tau = \sum_{b_j} \sum_{f_i} \tau_{b_j \leftarrow f_i}. \tag{3}$$

We adopt the cubic kernel function [Monaghan 1992] for its second-order continuity, which is used later in gradient computation, and assume the surfaces of rigid bodies are frictionless, therefore, no fluid frictional force at the fluid-rigid boundary is involved.

4.3 Rigid Body Dynamics & SPH-based Rigid Contact

For rigid body dynamics, we consider a semi-implicit time-stepping scheme as follows, where unit quaternion is adopted as the description of rigid body orientation:

$$\begin{aligned} v^{n+1} &= v^n + \Delta t \mathrm{M}^{-1} f^n, \\ x^{n+1} &= x^n + \Delta t v^{n+1}, \\ \omega^{n+1} &= \omega^n + \Delta t (\mathrm{I}^n)^{-1} (L^n \times \omega^n + \tau^n), \\ q^{n+1} &= \text{normalize}(q^n + \frac{\Delta t}{2}([0, \omega^{n+1}] \otimes q^n)), \end{aligned} \tag{4}$$

where Δt is the time step, M a positive diagonal mass matrix, f and τ the external force and torque, I and $L = \mathrm{I} \cdot \omega$ the inertia tensor and angular momentum of rigid body, respectively, and $[0, \omega]$ is a pure quaternion with imaginary part $\omega \in \mathbb{R}^3$. The normalization operation for q is to eliminate the numerical error to ensure the quaternion stays on the unit S^3 sphere.

For rigid-rigid contact, inspired by [Gissler et al. 2019; Xu et al. 2021a,b], we present a penalty-based SPH-based rigid-rigid contact model for its good differentiability, where the collision detection is done using SPH density formulation, and the normal contact force and friction force at rigid particle r is computed as:

$$F_r^{\text{normal}} = k(\frac{\rho_r}{\rho_0} - 1)\mathbf{n}_r, \quad F_r^{\text{friction}} = -\mu\|F^{\text{normal}}\|\frac{\mathbf{v}_\mathbf{r}^{\text{rel}}}{\|\mathbf{v}_\mathbf{r}^{\text{rel}}\|}, \quad (5)$$

where k, μ is the contact stiffness and friction coefficent, respectively. ρ_r is the SPH density computed only between rigid particles, $\mathbf{n}_r = \mathbf{x}_r - \frac{\sum_s \mathbf{x}_s W_{rs}}{\sum_s W_{rs}}$ is the estimated normal at r with s being the neighboring rigid particles of the same rigid body of r, and $\mathbf{v}_r^{\text{rel}} = \mathbf{v}_r - \frac{\sum_k \mathbf{v}_k W_{rk}}{\sum_k W_{rk}}$ is the estimated relative velocity at r with k being the neighboring rigid particles of the other rigid body of r.

5 DIFFERENTIABLE SPH-BASED FLUID-RIGID COUPLING

We now describe in detail how we differentiate the forward simulation introduced in Sec. 4. A comprehensive overview of the computational graph of the forward simulation and gradient computation for SPH-based fluid-rigid coupling is presented in Fig. 3. We adopt a forward-mode differentiation scheme, and the comparison with reverse-mode differentiation and the discussion of the reason for our choice is in Sec. 7.4.3.

We start by differentiating the semi-implicit integration of rigid body dynamics in Sec. 5.1 (shown as the gray block in Fig. 3(a)). Then we present a general differentiable formulation of SPH-based fluid-rigid coupling in Sec. 5.2, where we analyze the workflows of SPH-based fluid and solid forward simulations and extract a combined gradient computation scheme for the two-way coupled simulation. However, there will be quick gradient explosion issues with this naive general differentiation scheme, so we investigate possible causes of the gradient instability in Sec. 5.3 and finally present a stable and efficient localized gradient computation scheme in Sec. 5.4.

5.1 Gradients of Rigid-body Dynamics

According to the problem formulation introduced in Sec. 3, we need to compute the gradient of rigid body state $\mathbf{s}_\mathcal{R}^{n+1}$ at time step $n+1$ with respect to the initial state variable $\mathbf{s}_\mathcal{R}^0$ at time step 0 in a simulation trajectory, where $s^0 \in \mathbf{s}_\mathcal{R}^0$ can represent v^0, ω^0, x^0, q^0, etc. Then the semi-implicit integration of rigid body dynamics is differentiated from the chain rule as follows:

$$\begin{aligned}
\frac{\mathrm{d}v^{n+1}}{\mathrm{d}s^0} &= \frac{\mathrm{d}v^n}{\mathrm{d}s^0} + \Delta t \mathrm{M}^{-1}\frac{\mathrm{d}f^n}{\mathrm{d}s^0}, \\
\frac{\mathrm{d}x^{n+1}}{\mathrm{d}s^0} &= \frac{\mathrm{d}x^n}{\mathrm{d}s^0} + \Delta t \frac{\mathrm{d}v^{n+1}}{\mathrm{d}s^0}, \\
\frac{\mathrm{d}\omega^{n+1}}{\mathrm{d}s^0} &= \frac{\mathrm{d}\omega^n}{\mathrm{d}s^0} + \Delta t \left[\frac{\mathrm{d}(\mathrm{I}^n)^{-1}}{\mathrm{d}s^0}(L^n \times \omega^n + \tau^n)\right. \\
&\qquad \left. +(\mathrm{I}^n)^{-1}(\frac{\mathrm{d}(L^n \times \omega^n)}{\mathrm{d}s^0} + \frac{\mathrm{d}\tau^n}{\mathrm{d}s^0})\right], \\
\frac{\mathrm{d}q^{n+1}}{\mathrm{d}s^0} &= \frac{\mathrm{d}q^{n+1}}{\mathrm{d}\hat{q}}\left[\frac{\mathrm{d}q^n}{\mathrm{d}s^0} + \frac{\Delta t}{2}\frac{\mathrm{d}([0, \omega^{n+1}] \otimes q^n)}{\mathrm{d}s^0}\right],
\end{aligned} \quad (6)$$

where $\hat{q} := q^n + \frac{\Delta t}{2}([0, \omega^{n+1}] \otimes q^n)$ is the updated quaternion before the normalization operation. Note that the inertia tensor I is the function of the spatial orientation of rigid bodies since $\mathrm{I}^n = R^n \mathrm{I}^0 R^{n\top}$ with $R^n = R^n(q^n)$ as the rotation matrix, so we also need to take the derivative of I into account. For more details on gradient computation, we refer readers to the supplementary.

In Eq. (6), there are two unknowns $\frac{\mathrm{d}f^n}{\mathrm{d}s^0}, \frac{\mathrm{d}\tau^n}{\mathrm{d}s^0}$. Because in our control task, we focus on the state $\mathbf{s}_\mathcal{R}$ of rigid body $\mathcal{R}$ in a coupled fluid-solid system, we suppose the external net force and torque f^n, τ^n applied to $\mathcal{R}$ from the fluid environment at time step n as the function of $\mathbf{s}_\mathcal{R}^n$, since the change of $\mathbf{s}_\mathcal{R}^n$ will directly change the status of fluid-rigid interaction thus affecting f^n, τ^n. Therefore, we have $f^n := f^n(\mathbf{s}_\mathcal{R}^n) = f^n(v^n, x^n, \omega^n, q^n)$ and $\tau^n := \tau^n(\mathbf{s}_\mathcal{R}^n) = \tau^n(v^n, x^n, \omega^n, q^n)$. Then we solve $\frac{\mathrm{d}f^n}{\mathrm{d}s^0}$ ($\frac{\mathrm{d}\tau^n}{\mathrm{d}s^0}$ the same way) based on the chain rule:

$$\frac{\mathrm{d}f^n}{\mathrm{d}s^0} = \frac{\partial f^n}{\partial x^n}\frac{\mathrm{d}x^n}{\mathrm{d}s^0} + \frac{\partial f^n}{\partial v^n}\frac{\mathrm{d}v^n}{\mathrm{d}s^0} + \frac{\partial f^n}{\partial q^n}\frac{\mathrm{d}q^n}{\mathrm{d}s^0} + \frac{\partial f^n}{\partial \omega^n}\frac{\mathrm{d}\omega^n}{\mathrm{d}s^0}. \quad (7)$$

From Eq. (7), given $\frac{\mathrm{d}x^n}{\mathrm{d}s^0}, \frac{\mathrm{d}v^n}{\mathrm{d}s^0}, \frac{\mathrm{d}q^n}{\mathrm{d}s^0}, \frac{\mathrm{d}\omega^n}{\mathrm{d}s^0}$ from the last time step, we need to compute the partial derivatives of f^n, τ^n with respect to $s^n \in (v^n, \omega^n, x^n, q^n)$, which is related to the details of SPH-based fluid-rigid coupling and is introduced in the next subsection.

5.2 A General but Unstable Differentiation Scheme of SPH-based Two-way Fluid-Rigid Coupling

In this subsection, we introduce a general way to compute the gradient of SPH-based two-way fluid-rigid coupling. This scheme, however, causes instability issues such as quick gradient explosion in practice, which motivates us to investigate the causes and leads to the improved differentiation scheme introduced in later sections.

As introduced in Sec. 4.2, we discretize the surface of solids using boundary particles. In this model, the state $\mathbf{s}_{b_j}$ of each boundary particle b_j is represented by its position x_{b_j} and linear velocity v_{b_j}, namely, $\mathbf{s}_{b_j} = (x_{b_j}, v_{b_j})$, which are computed from the position x and rotation $R = R(q)$ of rigid body as: $x_{b_j}^n = x^n + r_{b_j}^n$ and $v_{b_j}^n = v^n + \omega \times r_{b_j}^n$ with $r_{b_j}^n = R^n r_{b_j}^0$, where r_{b_j} is the particle's displacement relative to the body's center of mass. Based on Eqs. (3) and (7), we compute the gradient of the particle-pair fluid-rigid coupling forces and torques on the rigid particles, and then aggregate these gradients to obtain the final gradient for the rigid body as a whole:

$$\frac{\partial f^n}{\partial s^n} = \sum_{b_j}\sum_{f_i}\frac{\partial F_{b_j \leftarrow f_i}}{\partial s^n} = \sum_{b_j}\sum_{f_i}\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}}\frac{\partial \mathbf{s}_{b_j}}{\partial s^n}. \quad (8)$$

To compute $\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}}$ in Eq. (8), we formally summarize the forward computational graph for SPH-based fluid-rigid coupling in one time step, as Fig. 3(a) shows. Both the states of rigid body particles and fluid particles contribute to the computation of coupling forces and torques. In Fig. 3(b), b_j, f_i, f_j denote the rigid particle (gray color), the fluid particle (blue color) neighboring b_j, and the fluid particle neighboring f_i, respectively. In the iterative solving process of fluid pressure projection, the states of b_j contribute to the update of the velocities of all fluid particles through the intermediate result of pressure forces, which affects the velocities of neighboring f_i and

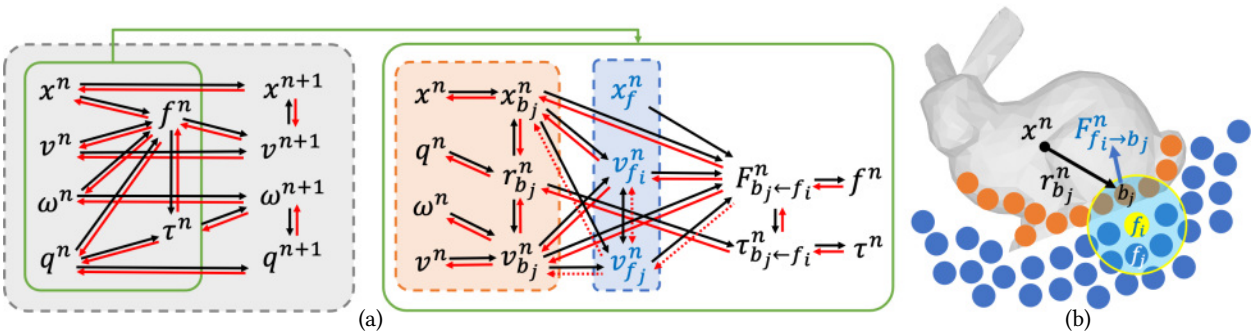


Fig. 3. The computational graph (a) and illustration (b) of SPH-based fluid-rigid coupling force and torque in the n^{th} time step. (a): In the gray dashed block on the left, the reliance between variables of solid motion in one-step integral is demonstrated. The green block part is analyzed in more detail on the right, where relationships between the rigid-body variables are shown in the orange dashed block, and fluid variables are contained in the blue dashed block. Variables with a subscript stand for those being near the solid-fluid boundary. Black arrows stand for forward calculation reliance, and red arrows stand for gradient computation reliance in our general differentiable formulation. (b): When b_j enters the support radius of the kernel function of f_i (the blue circle with yellow outline centered at f_i), a fluid-rigid coupling force is applied on b_j.

further influences the velocities of f_j even if f_j is not directly adjacent to b_j. We compute the gradient of this forward computational graph in Fig. 3(a):

$$\begin{aligned}\frac{\mathrm{d}F_{b_j\leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}} &= \frac{\partial F_{b_j\leftarrow f_i}}{\partial \mathbf{s}_{b_j}} + \frac{\partial F_{b_j\leftarrow f_i}}{\partial v_{f_i}}\frac{\mathrm{d}v_{f_i}}{\mathrm{d}\mathbf{s}_{b_j}} + \sum_{f_j}\frac{\partial F_{b_j\leftarrow f_i}}{\partial v_{f_j}}\frac{\mathrm{d}v_{f_j}}{\mathrm{d}\mathbf{s}_{b_j}} \\ &= \frac{\partial F_{b_j\leftarrow f_i}}{\partial \mathbf{s}_{b_j}} + \frac{\partial F_{b_j\leftarrow f_i}}{\partial v_{f_i}}\frac{\mathrm{d}v_{f_i}}{\mathrm{d}\mathbf{s}_{b_j}} \\ &\quad + \sum_{f_j}\frac{\partial F_{b_j\leftarrow f_i}}{\partial v_{f_j}}\left(\frac{\partial v_{f_j}}{\partial \mathbf{s}_{b_j}} + \sum_{f_p}\frac{\partial v_{f_j}}{\partial v_{f_p}}\frac{\mathrm{d}v_{f_p}}{\mathrm{d}\mathbf{s}_{b_j}}\right),\end{aligned} \tag{9}$$

where f_p is the neighboring particle of f_j, and we can further expand $\frac{\mathrm{d}v_{f_p}}{\mathrm{d}\mathbf{s}_{b_j}}$ to a formula contain $\frac{\mathrm{d}v_{f_i}}{\mathrm{d}\mathbf{s}_{b_j}}$. With the relationship between particle accelerations and coupling forces, as the iterative solving process goes on, k^{th}-order neighboring fluid particles are taken into account, leading to a recursive gradient computation formulation.

In Eq. (9), the gradient of the fluid-rigid coupling force $F_{b_j\leftarrow f_i}$ with respect to the state of rigid particle $\mathbf{s}_{b_j}$ and velocity of fluid particle v_{f_i} can be explicitly derived from Eqs. (1) (2), e.g.:

$$\begin{aligned}\frac{\partial F_{b_j\leftarrow f_i}}{\partial x_{b_j}} &= m_{f_i}V_{b_j}\left(\frac{1}{\rho_{f_i}}\nabla W_{ij}^{\top}\frac{\partial\left(\frac{p_{f_i}}{\rho_{f_i}}\right)}{\partial x_{b_j}} + \frac{p_{f_i}}{\rho_{f_i}}\nabla^2 W_{ij}\right), \\ \frac{\partial F_{b_j\leftarrow f_i}}{\partial v_{b_j}} &= -m_{f_i}V_{b_j}\left(\frac{1}{\rho_{f_i}}\nabla W_{ij}^{\top}\frac{\partial p_{f_i}}{\partial v_{b_j}}\right),\end{aligned} \tag{10}$$

with more details in the supplementary.

Finally, by assuming that for all fluid particle f, at the beginning of the iteration $\frac{\mathrm{d}v_f}{\mathrm{d}\mathbf{s}_{b_j}} = \mathbf{0}$, we recursively solve $\frac{\mathrm{d}F_{b_j\leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}} = -\frac{\mathrm{d}F_{f_i\leftarrow b_j}}{\mathrm{d}\mathbf{s}_{b_j}}$ with Eqs. (8)(9). Then we accumulate the gradient in all iterations to get the final gradient.

Eqs. (8)(9) will serve as a general differentiable formulation of the two-way coupled SPH-based simulation which leads to a recurrence formula to solve $\frac{\mathrm{d}F_{b_j\leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}}$. and the gradient of torque is $\frac{\mathrm{d}\tau^{(k)}_{b_j\leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}} = [r_{b_j}]\frac{\mathrm{d}F^{(k)}_{b_j\leftarrow f_i}}{\mathrm{d}\mathbf{s}_{b_j}} + [F^{(k)}_{b_j\leftarrow f_i}]^{\top}\frac{\mathrm{d}r_{b_j}}{\mathrm{d}\mathbf{s}_{b_j}}$, where $[r]\in\mathbb{R}^{3\times3}$ is the skew-symmetric matrix of $r\in\mathbb{R}^3$.

This computational scheme, in essence, is to evaluate the disturbance of the iterative solving process of fluid pressure projection with respect to the perturbation of rigid body boundary conditions. As iteration index k increases, to update the velocity v_{f_i} of fluid particle f_i, k^{th}-order neighboring fluid particles of the rigid body are involved, then the recursive solving of the fluid-rigid coupling gradient gradually accumulates the gradient contribution from the k^{th}-order of fluid particles.

In practice, however, we find that directly applying this gradient in optimization can cause instability issues, such as quick gradient explosion after gradient accumulation with DFSPH iterations in one time-step after fluid-rigid contact. A similar gradient explosion is also observed using the forward-mode differentiation of DiffTaichi [Hu et al. 2020]. So we consider this unstable gradient problem as a common issue in differentiating SPH-based fluid-rigid coupling simulation, which is a sign of the underlying non-smoothness of the system. This presents a significant challenge that hinders the differentiability of an SPH-based fluid-rigid coupling simulator, and we discuss the possible causes in the following subsection.

5.3 Instability in Differentiating SPH-based Two-way Fluid-rigid Coupling

There are two possible causes contributing to the unstable gradient result with the naive general differentiation scheme introduced above: the underlying chaos in differentiating the high-DoF particle interactions, and the sensitivity of SPH kernel function to particle distances.

5.3.1 Instability in differentiating the high-DoF particle representation. Mathematically, the recursive nature of Eqs. (8)(9) incurs a high risk of gradient explosion if any term that is repeatedly multiplied in the formula has an eigenvalue bigger than 1. Here we plot the magnitude of each term in Eq. (9) with respect to the iteration k in Fig. 4. It will be hard to control the eigenvalues of the terms

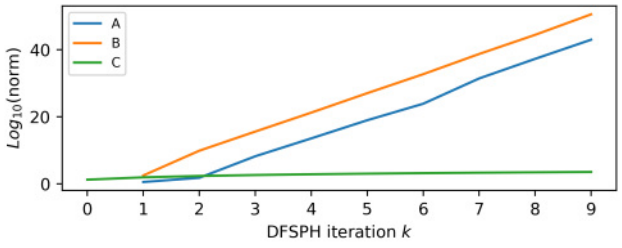
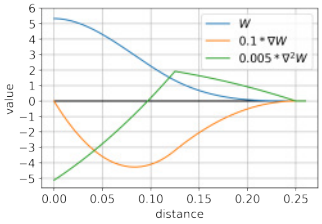


Fig. 4. The log plot of the norm of each term in Eq. (9) with respect to the iteration number k in our test scene similar to the water rafting scene in Fig. 8, where A, B, and C represent the terms $\frac{\partial F_{b_j \leftarrow f_i}}{\partial v_{f_i}} \frac{\mathrm{d}v_{f_i}}{\mathrm{d}s_{b_j}}$, $\sum_{f_j} \frac{\partial F_{b_j \leftarrow f_i}}{\partial v_{f_j}} \frac{\mathrm{d}v_{f_j}}{\mathrm{d}s_{b_j}}$ and $\frac{\partial F_{b_j \leftarrow f_i}}{\partial s_{b_j}}$ respectively.

A, B in the figure to always stay within a safe range by tweaking the overall scale of the physical system, which we left as future work. This quick gradient explosion issue can be related to the chaotic nature of the high-DoF particle interactions in DFSPH, where trying to figure out the sensitivity of each particle pair's interaction during the iterative solving process to input parameters leads to a combinational explosion.

5.3.2 Instability in differentiating SPH kernel function. We have observed that another possible contributing factor to the gradient instability is the high sensitivity of spatial derivatives of SPH kernel functions to the positions of solid boundary particles. Though the commonly used cubic kernel function W in SPH is second-order differentiable, however, the order of magnitude of its first and second-order derivatives $\nabla W, \nabla^2 W$ increases rapidly when the distance of two particles decreases, as shown in the scaled plot below.



When the fluid system is already at a state where the incompressibility constraint has been satisfied, perturbations in the position of the solid boundary can cause rapid changes in certain estimated fluid physical properties such as density, as a result of the sensitivity of the derivatives of the kernel function W to particle distances. This can sometimes introduce abrupt penalty forces from the fluid incompressibility constraint, leading to sudden changes in the fluid-rigid coupling forces and unstable gradients.

Based on these analysis, we propose several techniques to stabilize the gradient while achieving efficiency in the next subsection.

5.4 A Stable & Efficient Localized Differentiation Scheme

To tackle the aforementioned possible causes of non-smoothness in differentiating SPH-based fluid-rigid coupling, we first propose a localized gradient computation scheme to reduce the scale of the particle system in differentiation in response to the first possible cause of non-smoothness. Then we propose a reduced gradient computation scheme to handle the high sensitivity of SPH kernel functions to particle distances.

5.4.1 Localized Gradient Formulation of Fluid-Rigid Coupling. we first design our approach to reduce the gradient instability from the idea of reducing the scale of the problem based on a key assumption. Our key assumption here is that in fluid-solid coupling, small perturbations of the states of solid objects generally do not significantly affect the bulk motion of the fluid environment. One reason is that such perturbation only affects the state of the fluid surrounding the solid boundary at the next time step, and it takes time to propagate this perturbation to the further area of the fluid environment that not directly adjacent to solid boundaries due to incompressibility. Therefore, to control the state of rigid bodies interacting with the fluid environment, we propose to only consider the fluid surrounding the solid objects when computing the gradient of fluid-rigid coupling. In practice, we drop the unstable gradient contributed by the $k^{\text{th}}(k > 1)$-order neighboring fluid particles in the DFSPH pressure solver. Compared with Eq. (9), this method simplifies the original computational graph of coupling forces and torques by ignoring some of the gradient computations (dotted red arrows) in Fig.3(a), where only the fluid particles f_i adjacent to rigid body particles are considered in gradient computation:

$$\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}} \approx \frac{\partial F_{b_j \leftarrow f_i}}{\partial s_{b_j}} + \frac{\partial F_{b_j \leftarrow f_i}}{\partial v_{f_i}} \frac{\mathrm{d}v_{f_i}}{\mathrm{d}s_{b_j}}. \tag{11}$$

To solve $\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}}$, based on the impulse-momentum theorem:

$$\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}} \Delta t = -m_{f_i} \frac{\mathrm{d}v_{f_i}}{\mathrm{d}s_{b_j}}. \tag{12}$$

Based on Eqs. (11)(12), we derive:

$$\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}} = \frac{\partial F_{b_j \leftarrow f_i}}{\partial s_{b_j}} - \frac{\Delta t}{m_{f_i}} \frac{\partial F_{b_j \leftarrow f_i}}{\partial v_{f_i}} \frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}}, \tag{13}$$

from which, we solve $\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}}$ as the following compact formula, with $I \in \mathbb{R}^{3\times 3}$ being the identity matrix:

$$\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}s_{b_j}} = \left(I + \frac{\Delta t}{m_{f_i}} \frac{\partial F_{b_j \leftarrow f_i}}{\partial v_{f_i}} \right)^{-1} \frac{\partial F_{b_j \leftarrow f_i}}{\partial s_{b_j}}. \tag{14}$$

This leads to a localized gradient computation, which is also efficient since it allows us to obtain local information about the fluid while avoiding the high computational cost associated with differentiating the entire high-DoF system. With the warm start techniques introduced in [Bender and Koschier 2015] the number of required iterations to be unrolled per time step is not large in practice. We note that our assumption of a localized influence of rigid objects to fluid in fluid-solid coupling has bias from physical ground truth, however, we find it effective in optimization of our inverse control tasks of a range of complexity including Figs. 5,7,8,9, etc.

5.4.2 Reduced Gradient Computation Scheme. Based on the observation of the sensitivity of SPH kernel functions to particle distances, we propose to avoid directly perturbing the spatial position and rotation of the rigid body in gradient computation, and instead compute

the gradient only from perturbations in rigid body velocities. This leads to a reduced gradient formula from Eq. (7) presented below:

$$\frac{\mathrm{d}f^n}{\mathrm{d}s^0} = \frac{\partial f^n}{\partial v^n}\frac{\mathrm{d}v^n}{\mathrm{d}s^0} + \frac{\partial f^n}{\partial \omega^n}\frac{\mathrm{d}\omega^n}{\mathrm{d}s^0}, \tag{15}$$

which means we only compute $\frac{\mathrm{d}F_{b_j \leftarrow f_i}}{\mathrm{d}v_{b_j}}$ in the computational graph shown in Fig. 3(a). In practice, this reduced gradient leads to a more stable gradient result compared with using full gradients as Eq. (7), which achieves a more consistent update direction of design variables and achieves better optimization results in our experiments as discussed in Sec. 7.4.1.

5.4.3 Gradient of Neighborhood Search. To backpropagate through the neighborhood search operation in SPH, we assume that the set of neighboring fluid particles surrounding a rigid body particle does not experience large deviation between adjacent time steps, which is reasonable in most cases in our experiments as the local fluid region around a rigid body is generally subject to continuous, rather than abrupt, fluid movement. This assumption is also adopted in other research literature such as [Solenthaler and Pajarola 2009]. As a result, we utilize the same set of neighboring fluid particles to compute the gradients.

5.4.4 Integration of Differentiable Fluid-Rigid & Rigid-Rigid Coupling. The penalty-based SPH-based rigid-rigid contact model presented in Sec.4.3 can be readily differentiated, with more details in the supplementary. We integrate the gradient computation of this unified SPH-based coupling system by first computing the gradient of fluid-rigid coupling then followed by computing the gradient of rigid-rigid coupling based on chain rules.

In the preceding subsections, we have derived the gradients necessary for building a differentiable two-way SPH-based fluid-rigid coupling simulator from end to end. We integrate this gradient computation scheme into the forward computation: after the forward pass of each time step, we start the gradient computation from Eq. (6), with the fluid-rigid coupling gradient computed from Eqs. (8), (14) and (15) with more computational details in the supplementary.

6 ALGORITHM & IMPLEMENTATION

In this section, we present the main algorithm for gradient computation and implementation details, which is based on the theoretical foundations outlined in the preceding sections.

Algorithm. We propose the algorithms for the overall simulation time step and the gradient computation of DFSPH solvers as Algs. 2. The loss function introduced in Sec. 3 can then be optimized with the gradient of the simulator. We develop our algorithm based on DFSPH and incorporate the gradient computation into the forward simulation of the pressure solvers of DFSPH.

Implementation. We implement our method with C++ and Python based on the SPlisHSPlasH library [Bender et al. 2022] and test on an AMD Ryzen 5 R5600X CPU (6 cores, 3.7 GHz) with 32GB memory. We use Eigen for matrix and vector operations, and use OpenMP for parallelization. In the simulation, we use adaptive time-stepping according to the CFL condition described in [Bender and Koschier 2017]. For the SPH interpolation, we use the cubic spline kernel [Monaghan 1992] with a support radius of four times the particle radius. The rest density of the fluids is 1000 kg/m^3 while the largest permissible density and divergence error is 0.05 % and 0.1%. [Bender and Koschier 2017] is used for simulating fluid viscosity with a coefficient of 0.1, and [Akinci et al. 2013] is adopted for surface tension with a coefficient of 0.2-0.5 in our experiments (there is no viscosity force and surface tension adhesion force between fluid and rigid bodies) For neighborhood search, we adopt the parallel method of [Ihmsen et al. 2011]. The contact stiffness is 10^5 and the friction coefficient is 0.7 in the SPH-based rigid-rigid coupling if multi-rigid contact is considered in the scene. For gradient-based optimizer, we use the SGD with default momentum as 0.5, Adam optimizer and the `ReduceLROnPlateau` learning rate scheduler from Pytorch [Paszke et al. 2019]. For gradient-free optimizer for comparison, we adopt the CMA-ES and (1+1)-ES methods from Nevergrad library [Rapin and Teytaud 2018]. We randomize the optimizers with the same random seed for result reproduction.

ALGORITHM 2: Simulation timestep with gradient computation

Input: Fluid state $\mathbf{s}_{\mathcal{F}}^n$ (with each fluid particle storing volume V, position x, velocity v, DFSPH factor k^{DFSPH}), rigid body state $\mathbf{s}_{\mathcal{R}}^n$, gradient $\frac{\mathrm{d}\mathbf{s}_{\mathcal{R}}^n}{\mathrm{d}\mathbf{s}_{\mathcal{R}}^0}$

Output: Fluid state $\mathbf{s}_{\mathcal{F}}^{n+1}$, rigid body state $\mathbf{s}_{\mathcal{R}}^{n+1}$, gradient $\frac{\mathrm{d}\mathbf{s}_{\mathcal{R}}^{n+1}}{\mathrm{d}\mathbf{s}_{\mathcal{R}}^0}$

Function `TimeStep`
- find particle neighborhoods
- compute fluid particle densities
- compute fluid particle DFSPH factors k^{DFSPH}
- $\mathbf{v}_f^* =$ `DivergenceSolverWithGradient`(k^{DFSPH})
- **parallel forall** *fluid particles* f_i **do**
  - $\mathbf{v}_{fi}(t+\Delta t) = \mathbf{v}_{fi}^*$
- compute fluid particle non-pressure forces $\mathbf{F}_{fi}^{\mathrm{adv}}(t)$
- adapt time step size Δt according to CFL condition
- **parallel forall** *fluid particles* f_i **do**
  - $\mathbf{v}_{fi}^* = \mathbf{v}_{fi} + \Delta t \mathbf{F}_{fi}^{\mathrm{adv}}/m_{fi}$
- $\mathbf{v}_f^* =$ `DensitySolverWithGradient`(k^{DFSPH})
- **parallel forall** *fluid particles* f_i **do**
  - $\mathbf{x}_{fi}(t+\Delta t) = \mathbf{x}_{fi}(t) + \Delta t \mathbf{v}_{fi}^*$
- compute rigid-rigid contact
- update rigid body state with Eq. (4)
- compute rigid body gradient $\frac{\mathrm{d}\mathbf{s}_{\mathcal{R}}^{n+1}}{\mathrm{d}\mathbf{s}_{\mathcal{R}}^0}$ with Eq. (6)

end

7 EXPERIMENTS & EVALUATIONS

In this section, we show the utility of our method in a range of versatile rigid body control tasks within a coupled fluid-solid system.

7.1 Rigid Body Trajectory Optimization

We first present the application of our differentiable fluid-rigid coupling simulator in the rigid body trajectory optimization tasks where we design five scenes: water bottle flip challenge, stone skipping, water rafting, high diving, and on-water billiards.

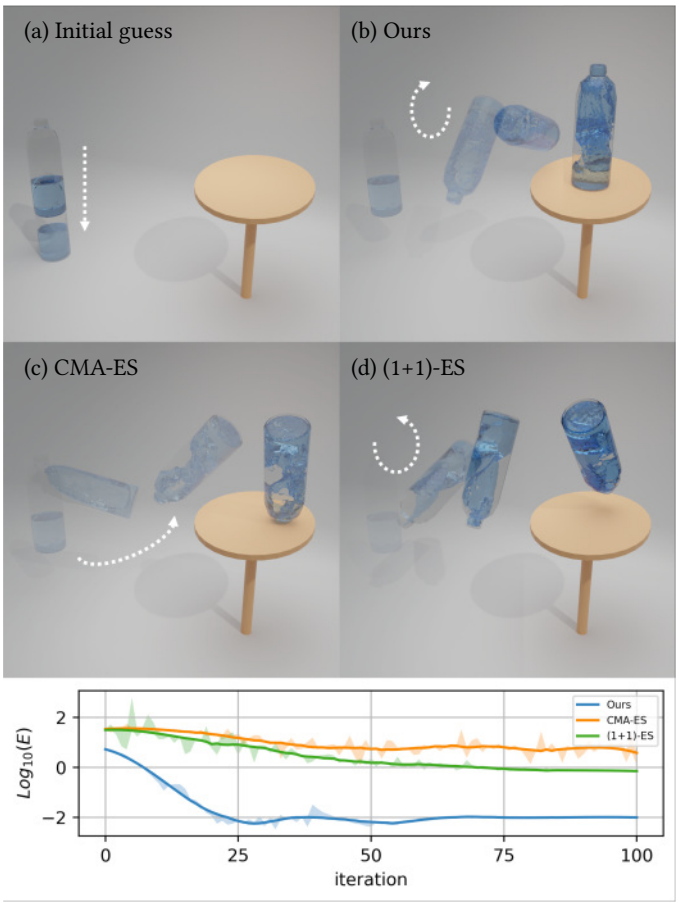


Fig. 5. Optimization results of different solvers of the first stage of water bottle flipping task to make the bottle land on the table without considering bottle-table collision with its base, namely, rotating 360 degrees. (a): Initial guess: The bottle freely falls to the ground with zero initial velocity. (b) Our optimization result with GD, where the bottle meets the goal well. (c)(d): CMA-ES and (1+1)-ES optimization results, the bottle doesn't fully meet the goal.

7.1.1 Water Bottle Flip Challenge. We start with the water bottle flipping task. As shown in Fig. 2, we define the design variables as the bottle's linear and angular velocity v^0, ω^0 at the moment of release, namely $\mathbf{p} = (v^0, \omega^0)$. The goal is to land on a user-specified target position x^* with the target orientation q^* at a user-specified time t^*. For example, to make the bottle land on its base, q^* is set to the corresponding orientation that makes the bottle rotate 360 degrees.

The interaction between the fluid and the bottle involves a large number of contacts and plays a key part in this task. As [Dekker et al. 2018] points out, a successful bottle flipping is owing to the exchange of angular momentum between the sloshing fluid and the bottle, which serves to decelerate the rotation of the bottle (Here we show the fast spin motion of flipping an empty bottle as a comparison in Fig. 6 (left) as a reference). Because the deformation of the bottle in the process is negligible, we treat the bottle as a rigid body. The collision between the bottle and the table is taken into account by combining the differentiability of this rigid-rigid contact and fluid-rigid coupling within our unified differentiable framework. Because the collision between the bottle and table gives an abrupt impulse to the motion of the fluid in the bottle, to make the optimization easier, we split the task into two stages: In the first stage, we only consider fluid-bottle coupling and optimize the bottle to reach the target position near above the table. Then in the second stage, we fine-tune the optimization result by considering the bottle-table collision and its coupling with fluid. The energy to be minimized in the first-stage optimization is defined as:

Fig. 6. (Left) Demonstration of fluid-rigid coupling: An empty bottle with the same optimized initial linear and angular velocity in Fig 5. Without fluid-rigid interaction, the bottle spins very fast. (Right) Based on the optimization result of the first stage where we only consider fluid-bottle coupling, we further consider the bottle-table collision stage to fine-tune the initial velocity of the bottle to make it able to stop and stand stably on the table.

$$\begin{aligned} \min_{\mathbf{p}} \quad & E = E_{\text{position}} + E_{\text{rotation}} \\ & E_{\text{position}} = w_{\text{position}} \|x^n - x^*\|^2 \\ & E_{\text{rotation}} = w_{\text{rotation}} \|q^n - q^*\|^2, \end{aligned} \tag{16}$$

with $w_{\text{position}}, w_{\text{rotation}}$ being the weight parameters. In the second stage, we drop the position loss to avoid over-constraint and only optimize the rotation loss to make the bottle stand stably on the stable at the end.

We show the first-stage optimized trajectories in Fig. 5 and compare the performance of one gradient-based solver gradient descent (GD) with momentum and learning rate scheduler, and two gradient-free solvers: CMA-ES and (1+1)-ES, which are standard gradient-free evolutionary strategies (ES) [Hansen 2006]. We set $w_{\text{position}} = 0.1, w_{\text{rotation}} = 1.0$ in experiments. We normalize the gradient direction for gradient-based optimization. We only manually set the initial learning rate as 1.0 for default unless stated for GD solver, and we adopt the learning rate scheduler to automatically reduce the learning rate once by half based on optimization performance. The patience of the scheduler is set as 5 for default unless stated. We can conclude from the results that the reduced gradient from our differentiable simulator facilitates gradient-based optimization to converge to the goal much faster than gradient-free optimizers.

For 3-dimensional rotation, there are multiple optimization paths to reach the target orientation (multiple optimization paths on the S^3 sphere of unit quaternion), with each path as a local minimum in the loss landscape. By setting different initial guesses, we are able to optimize the multiple optimization paths of rigid body rotations by our method. We show a different trajectory to achieve a 360-degree

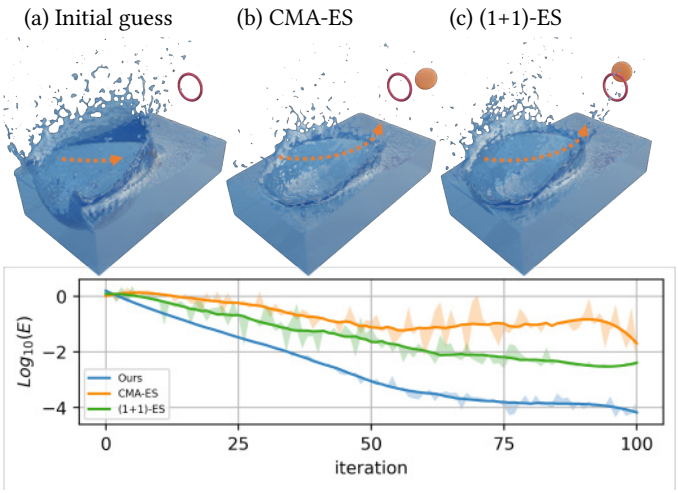


Fig. 7. Optimization results of the stone skipping task. (a): Initial guess: The stone hits water surface without bouncing. (b)(c): CMA-ES and (1+1)-ES optimization results at the 50$^{\text{th}}$ iteration. Our optimized result with GD at the 50$^{\text{th}}$ iteration are shown in Fig. 1. Our method achieves a much faster optimization speed.

rotation of the bottle in Fig. 6 (right) and use this initial velocity as a pre-trained result for second-stage optimization.

7.1.2 Stone Skipping. Our second task is stone skipping. Stone skipping, or stone skimming, is commonly observed in life as an entertainment activity, in which a flat stone is thrown across calm water at a high horizontal speed while spinning, and then it bounces off the water's surface due to lifting forces from fluid-rigid interactions. In this task, we define the design variables as the linear and angular velocity v^0, ω^0 of the stone at release, namely, $\mathbf{p} = (v^0, \omega^0)$. The goal is to make the stone reach a target position x^* at time t^* after bouncing off the water's surface.

Following the settings in [Nagahiro and Hayakawa 2005], we only consider the fluid-rigid coupling forces in the process without caring about the fluid surface tension. To avoid the case that the stone directly flies to the target without hitting and bouncing off the water, we add a constraint on the range of the vertical component of the initial linear velocity of the stone. Finally, the energy to be minimized is defined as:

$$\begin{aligned} \min_{\mathbf{p}} \quad & E = w_{\text{position}} \|x^n - x^*\|^2 + \lambda E_{\text{penalty}}, \\ & E_{\text{penalty}} = \mathbf{max}(v^0_{\text{vertical}} - v_{\text{thresh}}, 0), \end{aligned} \tag{17}$$

where $w_{\text{position},\lambda}$ are the weight parameters, v_{thresh} is the hyperparameter that upper-bounds the allowed initial vertical speed v^0_{vertical} of the stone.

We set $w_{\text{position}} = 1.0, \lambda = 1.0, v^0_{\text{vertical}} = -5$ in the experiment. We set the initial guess to be the state that the stone cannot successfully bounce off the water to let the optimizer explore the proper design variables to achieve the bounce. We show the initial guess and the optimized result at the 50$^{\text{th}}$ iterations given by GD with momentum and learning rate scheduler in Fig. 1 and compare the corresponding results of CMA-ES and (1+1)-ES solvers in Fig. 7.

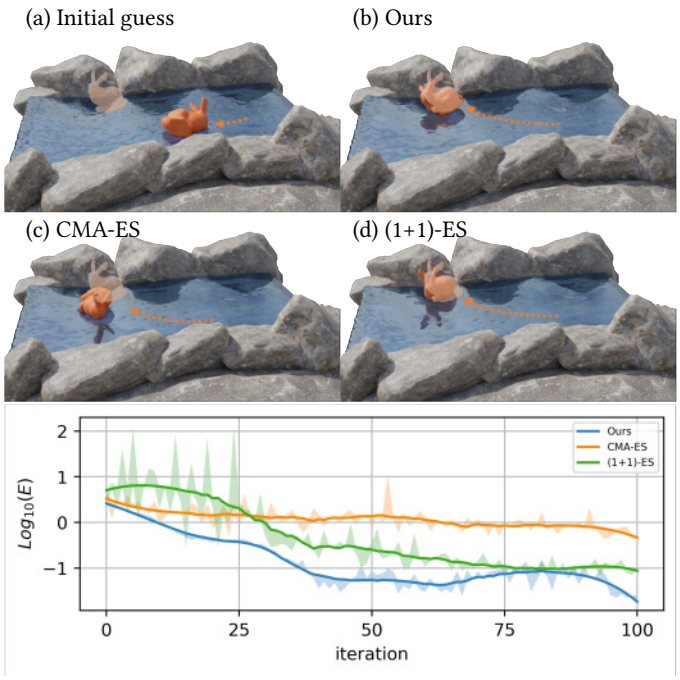


Fig. 8. Optimization results of different solvers of the water rafting task. (a): Initial guess: The bunny freely falls to the wave with zero initial velocity. The target pose is shown in yellow color. (b): Our optimization result with Adam, which is very close to the goal. (c)(d): CMA-ES and (1+1)-ES optimization results, the bunny doesn't fully meet the goal.

7.1.3 Water Rafting & High Diving. Our following two tasks involve the interaction of rigid agents with a large-scale dynamic fluid environment: the water rafting of a Stanford bunny, where the bunny needs to find the optimal initial linear and angular velocity to help itself to ride on the river flow to reach the target position and pose; and the high diving of a rigid duck, where it is required to perform optimal spin to achieve target rotation after rolling in the pool to make itself stand on the wave and face forward at the end (We do not consider rigid-rigid contact here). In the water rafting task, the design variables and goals are the same as the aforementioned water bottle flipping task, with the energy defined as Eq. (16) to make the bunny reach the target and rotate 180° in the vertical y-axis. In this scene, the fluid-rigid interaction strongly affects the rigid body's trajectory, and here we show a failure case as the figure below if we only intuitively optimize the vertical z-axis angular velocity and the linear velocity of the bunny.

In the high diving task, the design variable is only the initial angular velocity ω^0 of the rigid agent, namely, $\mathbf{p} = \omega^0$. The goal is to match the final orientation q^n with target orientation q^* at

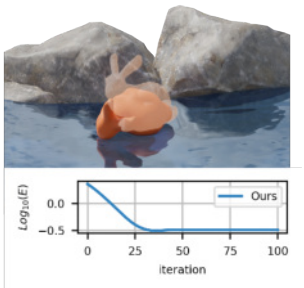


Fig. 10. Our gradient-based optimization fails to meet the goal with improper design variable settings.

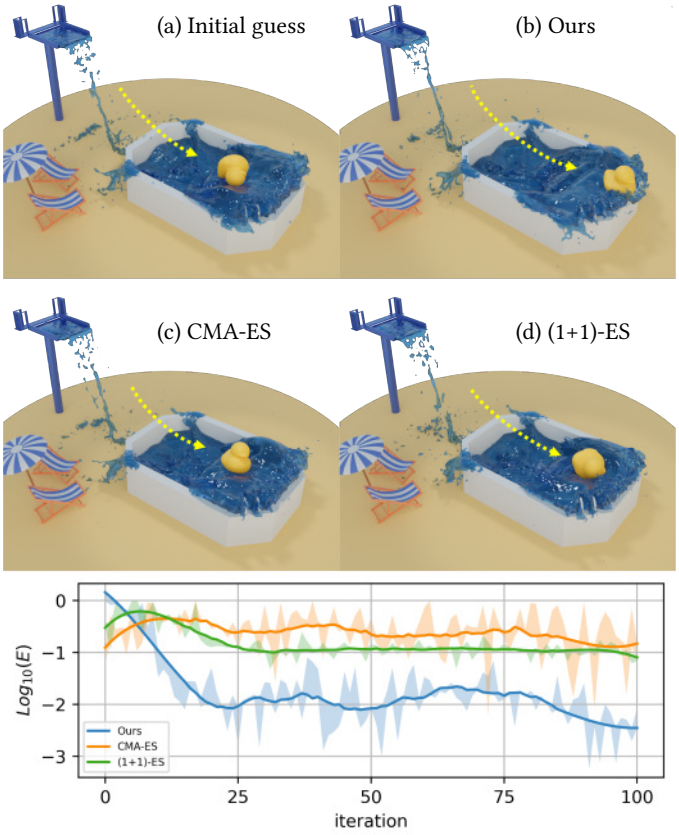


Fig. 9. Optimization results of different solvers of the high diving task to make the rigid duck reach target pose when entering the pool. (a): Initial guess: The duck freely falls to the flow with zero initial velocity. The target pose is to make the duck stand on the wave and face forward at the end (rotate 180° on the vertical axis). (b): Our optimization result with GD, where the duck matches the target standing straight pose the closest. (c)(d): CMA-ES and (1+1)-ES optimization results.

time t^* after rolling in the swimming pool, therefore, the energy to be minimized is defined as: $E = \|q^n - q^*\|^2$. In this scene, the duck rolls in the water pool, leading to a great influence of fluid dynamics on rigid motions with a large amount of fluid-rigid interactions. The results of these two tasks are shown in Figs. 8,9.

7.1.4 On-water Billiards. Finally, we design an on-water billiards scene where a yellow rigid ball hits a red ball in a water pool, and we optimize the initial linear and angular velocity of the yellow ball to make the red ball reach the target position after both rigid-rigid collision and fluid-rigid interaction. The energy is defined as $E = \|x^n_{\text{red ball}} - x^*\|^2$, and an end-to-end optimization is performed due to our unified differentiable fluid-rigid coupling simulator.

To sum up, the optimization results of all rigid body trajectory optimization tasks and simulation parameters in experiments are summarized in Tabs. 1 and 2. Considering that in some cases gradient-free methods can achieve good optimization results given enough

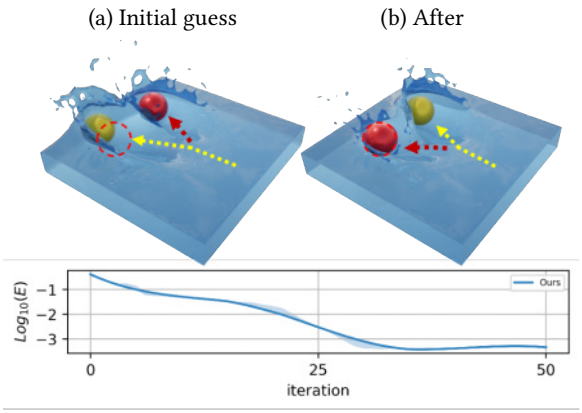


Fig. 11. Optimization result of on-water billiards scene using our method, where the target position of the red ball after the collision is highlighted with the dotted circle.

time, we discuss the advantage of our gradient-based method and compare it with gradient-free methods in Sec. 7.4.5. In addition, the loss difference at the outset in all loss curves is that we set the same initial design parameters for different methods but (1+1)-ES and CMA-ES will add a random offset to the design parameters based on their initial evolutionary strategies at the outset.

7.2 Self-supervised Learning of Water Bottle Flipping Control Policy

Learning-based methods that train neural network controllers with simulation data have also been an important research topic in various control applications. To demonstrate the applicability of our work in this direction, we present an example of the self-supervised learning of a water bottle flipping controller. We integrate our differentiable fluid-rigid coupling simulator into a neural network controller, which facilitates the training to be in an end-to-end manner with the gradient information directly from the simulator.

Specifically, in this task, we fix the initial position and orientation of the bottle, and the goal of this task is to control the release linear and rotational speed of the bottle to make it land on the target position x^* with a 360-degree rotation. x^* is given by users within a predefined 3-dimensional area $\mathcal{A}$. We minimize the same energy as in Eq. 16, despite setting $w_{\text{position}} = w_{\text{rotation}} = 1.0$. We set $\mathcal{A}$ a 3D rectangular area with size $[0, 5] \times [-1, 5] \times [0, 5]$.

We design the network architecture as three dense layers with 32 neurons in each layer and ReLU activations, as shown in Fig. 12. To generate the training and testing data, we randomly sample $N_{\text{train}} + N_{\text{test}}$ target positions in $\mathcal{A}$. We set $N_{\text{train}} = 1000$ and $N_{\text{test}} = 50$. We set the batch size as 5 and adopt the Adam optimizer [Kingma and Ba 2014] to train the network, with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ and a learning rate of 0.01 which is reduced by a factor of 0.5 after every epoch. We test the model every 50 batches. The convergence on training and test data of the learning process is shown in Fig. 12.

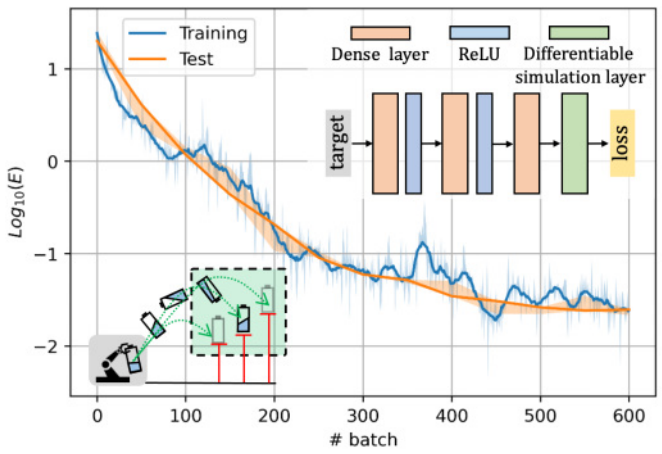


Fig. 12. (left): Illustration of this task where the gray block represents the neural network controller, and the green dashed block shows the predefined target area. (middle): Convergence on training and test data of the learning process for the water bottle flipping controller. (right): The corresponding controller network architecture.

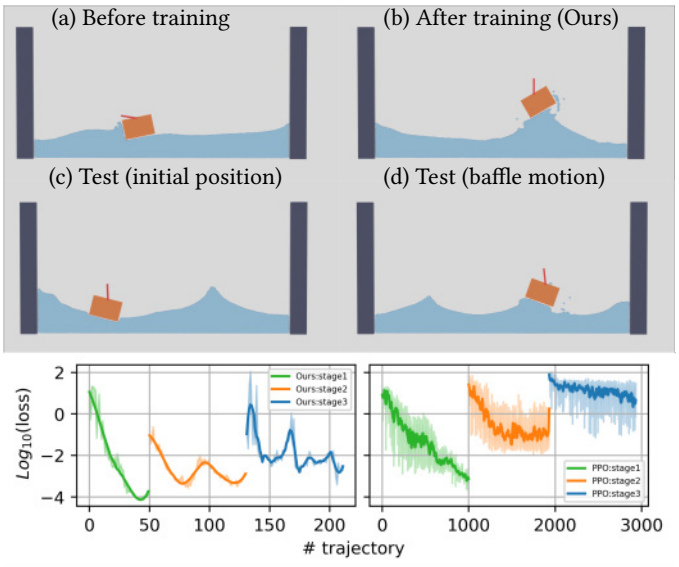


Fig. 13. Control results of the 2D on-water inverted pendulum robot on a choppy water surface disturbed by two moving baffles, with the control signal as an external acceleration applied to the cart every 50 time steps. (a): Before training, the pole quickly loses balance (b): After training with our differentiable simulator, the robot is able to succeed in the challenging water flows. (c)(d): We test the model trained with differentiable simulation with different initial positions and baffle motion parameters. The curve below shows the loss curve in training (the lower the better), where our gradient-based method is about an order of magnitude faster than PPO.

7.3 Closed-loop On-water Inverted Pendulum Robot Controller

In this example, we demonstrate the extensibility of our differentiable simulator to the two-way coupling of fluid and multi-body systems. We simulate a 2-dimensional on-water inverted pendulum robot with the impulse-based multi-body simulation method from [Bender and Schmitt 2006]. The robot consists of two rigid body parts: a pole and a cart, and the pole is attached by an unactuated joint to the cart. Different from previous open-loop tasks, the goal of this task is to train a closed-loop controller that keeps the pole balanced on the undulating water surface disturbed by two baffles, where the controller applies appropriate forces on the cart at regular intervals. The gradient of the multi-body simulation can be integrated into our differentiable fluid-rigid coupling framework, with computational details in the supplementary.

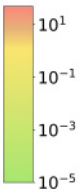
7.3.1 Experiment Settings. Specifically, we train a closed-loop control policy $\mathbf{a}_t = \phi_\theta(\mathbf{s}_t)$, which takes as input the current state $\mathbf{s}_t$ of the task and outputs an action vector $\mathbf{a}_t$ at time step t, and $\mathbf{a}_t$ is generated every N time steps. We represent the control policy ϕ_θ as a neural network parameterized by θ consisting of three dense layers with 16 neurons in each layer and ReLU activations at the end of the first two layers. The state vector $\mathbf{s}_t$ includes the position, rotation, linear and angular velocities of each part of the inverted pendulum robot. The action vector $\mathbf{a}_t$, or control signals, includes the horizontal and vertical accelerations applied to the cart, which are limited in ranges ($[-0.5/\Delta t, 0.5/\Delta t]$ for horizontal and $[-0.5/\Delta t, 0]$ for vertical). By integrating the policy network with our differentiable fluid-rigid coupling simulator, we are able to train the policy network in an end-to-end manner, with the loss function introduced below.

In each epoch during training, we simulate a trajectory with L time steps and apply the action $\mathbf{a}_t$ from the policy network every N time steps, which is set as 50 in our experiment. We can split the entire trajectory into many short horizons between two sequential actions, with each short horizon consisting of L/N time steps. In each short horizon, we define the loss function as $l = \frac{1}{2}\|q^n - q^*\|^2$, with q^n, q^* denoting the rotation of the pole at the end time step n of this short horizon, and the target rotation of the pole (which corresponds to the upright direction in this task), respectively. Then we backpropagate the short horizon with our differentiable simulator to get the gradient of the loss function with respect to the initial linear velocity v^0 of the cart at the beginning of the short horizon. The gradient of the loss is then used by a gradient-based optimizer Adam [Kingma and Ba 2014] to update the parameters of the policy network. In practice, we adopt the same three-stage training strategy to both methods: we first train the controller to keep the pole balanced for a 1s trajectory, then based on the trained model, we train the controller to succeed for a 3s trajectory, then based on the trained model to further succeed for a 5s trajectory. The reason for this curriculum learning strategy is that we find that compared with directly training the controller on a trajectory with total time as 5s, it is more reasonable to make the controller learn to succeed in easier tasks (e.g., keep the pole balanced from 0s to 1s) and then move to harder tasks (e.g., keep balanced from 0s to 5s).

To generate undulating water surfaces, we place two baffles on the left and right sides of the pool, and animate the baffles following a sine motion, with their amplitudes, periods, and initial phases fixed during training.

Table 1. Comparison between the performance of gradient-based and gradient-free optimization on all examples in Sec. 7. The "DoFs" and "#**P**" columns report the DoFs of the fluid-rigid system (6× rigid body numbers + 3× fluid particle numbers) and the number of design variables in the experiments, respectively. The "E^0" column reports the initial energy at the 0^{th} iteration. The "Optimized E Percentage" column reports the optimized E as a percentage (0-100%) of the initial energy E^0. The "Final Step" report the best results in each method when we end the optimization at the same number of iterations. The "50 Iters" columns report the best results in each method within 50 iterations. We color the values of energy percentage using a green-red color scale: green corresponds to 0% and red corresponds to 100%.

Task	DoFs	#**P**	E^0	Optimizated E Percentage					
				Ours		CMA-ES		(1+1)-ES	
				Final (%)	50 Iters(%)	Final (%)	50 Iters(%)	Final (%)	50 Iters(%)
Water Bottle Flip (Fig. 5)	39942	6	5.10	0.0019	0.0040	23.0034	23.0728	13.5363	22.4852
Stone Skipping (Fig. 7)	713103	6	1.36	0.0031	0.0510	0.4339	2.8831	0.1818	0.4887
Water Rafting (Fig. 8)	315468	6	2.32	0.9617	1.2778	18.5667	21.3904	2.8812	4.6056
High Diving (Fig. 9)	468372	3	0.97	0.0596	0.1332	0.6330	2.8138	4.9385	4.9385



7.3.2 Results and Comparisons. To compare our method with gradient-free methods, we also solve the task with the state-of-the-art reinforcement learning method PPO [Schulman et al. 2017]. For a fair comparison, we set the same policy network, state, and action for PPO, and set the reward in training to be the negative counterpart of l. The training results with the three-stage training strategy is shown in Fig. 13 with stage1, stage2, and stage3 representing training on 1s, 3s, and 5s trajectories, respectively. Compared with PPO, our method achieves an order of magnitude faster optimization speed.

We also test the generalizability of the trained policy network by our differentiable simulator to unseen fluid environments, where we change the motion parameters of baffles to generate different water flows from training and change the starting position of the robot. Furthermore, though we only train the controller to keep balanced from 0 to 5s, in the test it can keep the pole balanced for at least 9s. The simulation results are shown in Fig. 13, where our controller trained with differentiable simulation works well in challenging water flows.

7.4 Discussions & Evaluations

In this subsection, we provide ablation studies and discussions to evaluate the efficacy of the design choices made in our approach.

7.4.1 Reduced Gradient Scheme vs. Unstable Complete Gradient Scheme. We first compare the optimization performance of the reduced gradient computational scheme with the complete one introduced in Sec.5. We have observed the instability in computing the complete gradient of Eq. (7), based on which we propose a reduced gradient computational scheme Eq. (15) to alleviate the non-smoothness while obtaining plausible results. We compare the performance of the reduced and complete gradient computational scheme (Eqs. (7) and (15)) in the water bottle flip task as an example. In the experiment, the value of the complete gradient quickly comes to a large magnitude which causes the gradient to be unusable, as the results shown in Fig. 14, where our reduced gradient computational scheme gives plausible gradient directions and optimizes well. Note that the "plateau" here of the complete gradient is not a real plateau but fluctuates in the log-scaled plot. On the other hand, the learning rate is gradually reduced by the scheduler.

7.4.2 Discussion on DiffTaichi. We adopt DiffTaichi (version 1.5.0) to the Stanford Bunny water rafting scene. However, DiffTaichi turns out to be hard to accomplish such tasks. Due to the limitation of source code transformation [Hu et al. 2020], DiffTaichi does not work well in reverse-mode differentiation to handle the complex control flow including nested loops and branch statements of the DFSPH solvers that we use. Then we use DiffTaichi with forward-mode differentiation. In the forward mode, without our localized gradient technique, DiffTaichi will consider all the 100k+ fluid particles in the system and automatically compute the gradient along the forward simulation. As discussed in Sec. 5.3, directly differentiating the complete gradient from unrolling the DFSPH computational graph can lead to gradient explosion issues. In practice, we also observe that the gradient of fluid-rigid coupling forces generated by DiffTaichi quickly increases in magnitude as DFSPH solver iteration goes up in one time step after fluid-rigid contact happens.

7.4.3 Forward-mode Differentiation vs. Reverse-mode Differentiation. In this work, we differentiate the two-way fluid-rigid coupling system based on a forward-mode differentiation idea. To propose a reverse-mode differentiation scheme here is non-trivial, where the difficulty lies in that there is not an explicit first-order optimal (Karush-Kuhn-Tucker or KKT) system in the DFSPH solver and

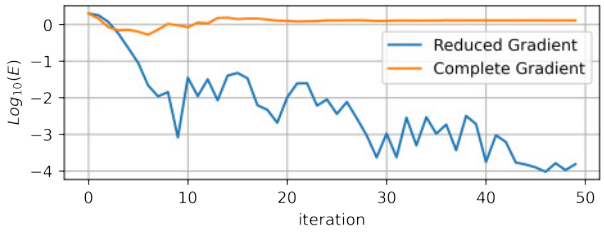


Fig. 14. Plot of loss function: Comparison of optimization results of the proposed reduced gradient scheme and the unstable complete gradient scheme in the water bottle flipping task.

Table 2. Overview of simulation parameters and experiment settings of all examples in Sec. 7 except the self-supervised learning water bottle flipping controller. $\#N_{\mathcal{F}}, \#N_{\mathcal{R}}$ denotes the number of fluid and dynamic rigid body particles in the simulation. $\rho_{\mathcal{R}}/\rho_{\mathcal{F}}$ reports the estimated average density ratio of the rigid body and the fluid. Δt is the average time step of the simulation. n is the average number of time steps when reaching the user-specified target time of one trajectory in optimization iterations. t_{forward} and t_{gradient} represents the average forwarding time and gradient computing time in a trajectory respectively. $\#N_{\text{gradient}}$ denotes the number of DoFs involved in gradient computation and $\#N_{\text{total}}$ denotes the total DoFs in the system.

Task	$\#N_{\mathcal{F}}$	$\#N_{\mathcal{R}}$	$\rho_{\mathcal{R}}/\rho_{\mathcal{F}}$	$\Delta t[s]$	n	$t_{\text{forward}}[s]$	$t_{\text{gradient}}[s]$	$\#N_{\text{gradient}}/\#N_{\text{total}}$
Water Bottle Flip (Fig. 5)	13312	14048	0.4	3.6×10^{-4}	2771	61.7	74.7	33.2%
Stone Skipping (Fig. 7)	237699	807	1.9	1.6×10^{-4}	1478	472.6	0.8	0.4%
Water Rafting (Fig. 8)	105154	1850	0.3	3.1×10^{-3}	635	125.1	3.9	0.9%
High Diving (Fig. 9)	153389	2733	0.3	7.4×10^{-4}	3092	688.0	6.5	0.7%
On-water Billiards (Fig. 11)	87374	2808	0.4	1.1×10^{-3}	226	29.3	0.5	0.9%
On-water Inverted Pendulum (Fig. 13)	2871	100	0.2	1×10^{-3}	-	-	-	-

how to formulate the adjoint state for gradient computation is unclear. As mentioned in Sec. 7.4.2, the reverse-mode differentiation of DiffTaichi also fails to compute the gradient of the DFSPH solver with fluid-rigid coupling. Furthermore, theoretically, reverse-mode differentiation incurs more memory cost than forward-mode differentiation since the latter computes the gradient alongside the forward simulation without the need for one additional backward pass. So we decide to leave the reverse-mode differentiation as a future work.

7.4.4 Failure Cases in Gradient-based Optimization. Sometimes, the gradient-based optimization can stuck in local minimums. Here we show one failure case in the water rafting task in the rigid bunny water rafting scene in Fig. 10, where we only optimize the vertical y-axis angular velocity and the linear velocity. However, the fluid-rigid interaction in this scene greatly affects the motion of the bunny, so this optimization setting may make the target unachievable, or the optimization is stuck at the local minimum.

7.4.5 Comparisons between Gradient-based and Gradient-free Methods. Compared with gradient-free methods such (1+1)-ES, though our method requires additional implementation effort, (1+1)-ES may take a long time to search for a usable result (which may still not fully meet the goal), while our gradient-based method uses much less time to meet the goal. This is demonstrated in Figs. 5, 7, 8 and 9. Furthermore, our method facilitates us to integrate our differentiable simulator into other neural networks for end-to-end training.

7.4.6 Comparison with finite difference. To further investigate the property of our method, we compare the gradient obtained from the finite difference method with ours. At the beginning of each time step, we manually perturb the velocity v of the rigid body by adding δv_i to the $i^{\text{th}}(i \in \{1,2,3\})$ entry of v, and compute the gradient of net fluid-coupling force f w.r.t. v_i as $\frac{\delta f}{\delta v_i} \in \mathbb{R}^3$, which is then used to assemble the final finite difference gradient $\frac{\delta f}{\delta v} \in \mathbb{R}^{3\times3}$. We take the stone skipping scene in Sec. 7.1.2 as the test case and use $\delta v_i = 10^{-6}, 10^{-8}, 10^{-10}$m/s, respectively. However, in practice, the finite difference gradient fails to keep stable, as Fig. 15 shows. We find that even with a very small perturbation of velocity, the change of the SPH-based fluid-rigid coupling force can be relatively big compared with the magnitude of δv_i, e.g., $\|f\| \sim$ 5k+N and $\|\delta f\| \sim$ 0.01N when $\delta v_i = 10^{-10} m/s$, causing $\|\frac{\partial f}{\partial v}\|$ to be bigger than 10^8. The possible reason can be related to the round-off numerical error in float32-precision SPH-based fluid simulation, which makes finite difference vulnerable and not able to always give valid gradients while our method demonstrates good stability.

7.4.7 Sensitivity of gradient calculation to the number of one-ring fluid particles. To investigate the stability of our localized gradient computation scheme with respect to particle resolution, we take the stone skipping scene in Sec. 7.1.2 for testing and adjust the particle radius to be 0.015(original), 0.012, 0.009, 0.006, 0.005, respectively, where the total number of fluid particles in the original scene will range from 200k+ to 6M+. We report the averaged norm of gradients of all time steps and repeat the experiment at each resolution 5 times. The result in Fig. 16 shows that the magnitude of the gradient contributed from the one-ring neighbor fluid particles is bounded, where the gradient only changes by $\sim$ 15% when the number of one-ring neighbor fluid particles is 5× more, so we believe particle resolution does not harm the stability of our method.

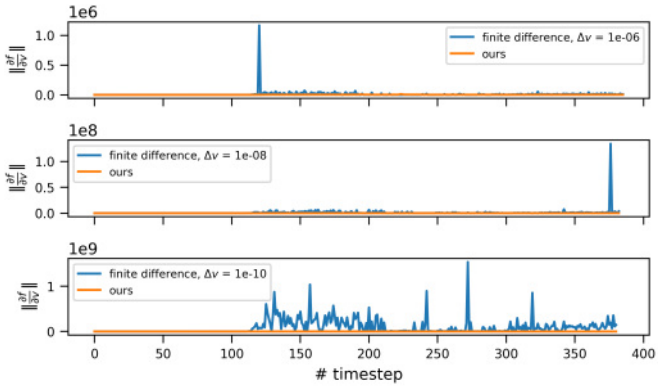


Fig. 15. Plot of the norm of gradient obtained from finite difference method with different δv_i and ours. The finite difference gradient fails to keep stable while our method demonstrates good stability.

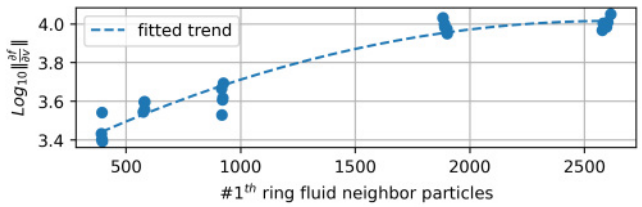


Fig. 16. Plot of the averaged norm of our gradient of fluid-rigid coupling force to rigid velocity $\frac{\partial f}{\partial v}$, where the magnitudes of gradients are bounded and increase slowly when particle resolution increases rapidly.

7.4.8 Influence of continuously varying initial conditions. Given the non-linear and non-convex nature of the optimization tasks, different initial guesses might converge to different local minimum results to achieve the goal for complex tasks. To investigate the sensitivity of our optimization results with the choice of different initial conditions, we choose to continuously vary the initial conditions of the bottle flip scene in Sec. 7.1.1 as a test case. We fix zero initial linear and angular velocities and sample 5 initial positions and rotations on a continuous parameterized curve as shown in Fig 17(top). We run the optimization with these different initial settings and record the value of the design parameters v^0, ω^0 along the optimization process to get their trajectories. The results in Fig 17 show that each initial setting converges to a solution to meet the same goal of reaching the same target position and a 360° degree rotation in the horizontal z-axis with the final loss less than 1e-2, and the optimization trajectories of v^0 change continuously with varying initial conditions. It is worth noting that when dealing with 3-dimensional rotation, there may be multiple optimization paths (local minimums) to achieve the desired rotation. As a result, it is observed that the optimized angular velocities for the 1st and 5th initial conditions differ from those of the other initial conditions with varying x-axis components while still meet the goal. Thus, We believe our optimization results are not overly sensitive to the choice of initial guesses from our experiments.

8 CONCLUSIONS

In this paper, we propose a novel differentiable particle-based two-way fluid-rigid coupling simulator for solving rigid body control tasks. To address the differentiability of the couple fluid-rigid system, we utilize the SPH methods to achieve a unified representation for both fluids and solids using particles, which makes the system easier to differentiate. To tackle the challenge of differentiability and high computational cost due to the high DoFs of fluid dynamics, after investigating the instability in gradient computation, we propose that it is not necessary to consider every detail of fluid motion but only take into account the fluid near the rigid boundaries in differentiating fluid-rigid coupling. This leads to a stable and efficient localized gradient computation scheme by focusing on the fluid particles that neighbor the rigid body particles. In practice, we also propose a reduced gradient formula for alleviating the instability in gradient computation. Our differentiable fluid-rigid simulator enables the use of gradient-based optimization methods in a diverse range of rigid body control tasks and demonstrates its efficacy compared

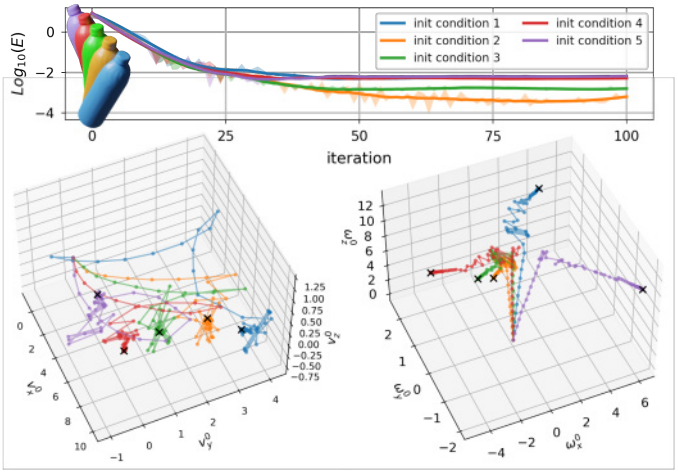


Fig. 17. (top): Five sampled continuously varying initial conditions of bottle flip and the corresponding optimization energy curve. Each initial setting converges to a solution to meet the same goal. (bottom): The 3-dimensional trajectories of design parameters v^0 and ω^0 in the optimization process of different initial conditions.

with gradient-free methods and reinforcement learning. We also demonstrate the extensibility of our method to the control of multi-body systems and multi-rigid contacts within a coupled fluid-rigid system. We believe our work will inspire further exploration in the field of differentiable simulation and the differentiable control of high-DoF systems.

Limitation and future work. There is some room to improve our work and other ideas that are worth further study. First, surface tension and boundary friction forces and their gradient can be considered in the fluid-rigid coupling simulation. Second, our method is based on DFSPH, which iteratively solves fluid pressure projection with density and divergence solvers, while another type of SPH method IISPH [Ihmsen et al. 2014] and pressure boundary method [Band et al. 2018a] solves the coupled fluid-solid system as a unified linear system. How to differentiate the solving of this large-scale linear system to build another type of particle-based fluid-rigid coupling simulator can be explored. Third, the differentiability of grid-based two-way fluid-rigid coupling is also an interesting research topic. We will also consider generalizing our trained bottle-flip controller and on-water inverted pendulum controller to other SPH methods. Fourth, It could be valuable to conduct a further theoretical investigation into the problem of gradient explosion in the naive general differentiation scheme. Another challenge would be to determine how to achieve a gradient direction that is closer to the ideal one while still maintaining computational efficiency. Last but not least, our experiments are made in the simulation environments, and we would like to further investigate the problems in closing the sim-to-real gap, e.g., the calibration between real fluid behavior with simulation parameters, and apply our method to real-world control tasks, e.g., a real-world robotic bottle flip.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work is supported by the National Key RD Program of China (2022YFB3303400), the National Natural Science Foundation of China (62025207, 62272245), the Fundamental Research Funds for the Central Universities (Nankai University) (63233080), and the Open Project Program of the State Key Lab of CADCG, Zhejiang University (Grant No. A2207).

REFERENCES

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Trans. Graph.* 32, 6, Article 182 (nov 2013), 8 pages. https://doi.org/10.1145/2508363.2508395

Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. 2012. Versatile rigid-fluid coupling for incompressible SPH. *ACM Trans. Graph.* 31, 4 (Aug. 2012), 1–8. https://doi.org/10.1145/2185520.2185558

Stefan Band, Christoph Gissler, Markus Ihmsen, Jens Cornelis, Andreas Peer, and Matthias Teschner. 2018a. Pressure Boundaries for Implicit Incompressible SPH. *ACM Trans. Graph.* 37, 2, Article 14 (feb 2018), 11 pages. https://doi.org/10.1145/3180486

Stefan Band, Christoph Gissler, Andreas Peer, and Matthias Teschner. 2018b. MLS pressure extrapolation for the boundary handling in divergence-free SPH. In *Proceedings of the 14th Workshop on Virtual Reality Interactions and Physical Simulations (VRIPHYS '18)*. Eurographics Association, Goslar, DEU, 55–63.

Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A Fast Variational Framework for Accurate Solid-Fluid Coupling. (2007).

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.

Jan Bender et al. 2022. *SPlisHSPlasH Library*. https://github.com/InteractiveComputerGraphics/SPlisHSPlasH

Jan Bender and Dan Koschier. 2015. Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. ACM, Los Angeles California, 147–155. https://doi.org/10.1145/2786784.2786796

Jan Bender and Dan Koschier. 2017. Divergence-Free SPH for Incompressible and Viscous Fluids. *IEEE Trans. Visual. Comput. Graphics* 23, 3 (March 2017), 1193–1206. https://doi.org/10.1109/TVCG.2016.2578335

Jan Bender, Tassilo Kugelstadt, Marcel Weiler, and Dan Koschier. 2019. Volume Maps: An Implicit Boundary Representation for SPH. In *Motion, Interaction and Games*. ACM, Newcastle upon Tyne United Kingdom, 1–10. https://doi.org/10.1145/3359566.3360077

Jan Bender and Alfred Schmitt. 2006. Fast Dynamic Simulation of Multi-Body Systems Using Impulses. In *Virtual Reality Interactions and Physical Simulations (VRIPhys)*. Madrid (Spain), 81–90.

Gilles Daviet and Florence Bertails-Descoubes. 2016. A Semi-Implicit Material Point Method for the Continuum Simulation of Granular Materials. *ACM Trans. Graph.* 35, 4, Article 102 (jul 2016), 13 pages. https://doi.org/10.1145/2897824.2925877

Pim J. Dekker, Lumen A. G. Eek, Mees M. Flapper, Remco H. J. C. Horstink, Anne R. Meulenkamp, Jelle van der Meulen, Stefan Kooij, Jacco H. Snoeijer, and Alvaro Marin. 2018. Water Bottle Flipping Physics. *American Journal of Physics* 86, 10 (Oct. 2018), 733–739. https://doi.org/10.1119/1.5052441 arXiv:1712.08271 [physics].

Ounan Ding and Craig Schroeder. 2020. Penalty Force for Coupling Materials with Coulomb Friction. *IEEE Transactions on Visualization and Computer Graphics* 26, 7 (2020), 2443–2455. https://doi.org/10.1109/TVCG.2019.2891591

Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. 2022. DiffPD: Differentiable Projective Dynamics. *ACM Trans. Graph.* 41, 2 (April 2022), 1–21. https://doi.org/10.1145/3490168

Tao Du, Kui Wu, Andrew Spielberg, Wojciech Matusik, Bo Zhu, and Eftychios Sifakis. 2020. Functional optimization of fluidic devices with differentiable stokes flow. *ACM Trans. Graph.* 39, 6 (Dec. 2020), 1–15. https://doi.org/10.1145/3414685.3417795

Yu Fang, Ziyin Qu, Minchen Li, Xinxin Zhang, Yixin Zhu, Mridul Aanjaneya, and Chenfanfu Jiang. 2020. IQ-MPM: an interface quadrature material point method for non-sticky strongly two-way coupled nonlinear solids and fluids. *ACM Trans. Graph.* 39, 4 (Aug. 2020). https://doi.org/10.1145/3386569.3392438

C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. 2021. Brax–A Differentiable Physics Engine for Large Scale Rigid Body Simulation. *arXiv preprint arXiv:2106.13281* (2021).

Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. 2020. ADD: analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Trans. Graph.* 39, 6 (Dec. 2020), 1–15. https://doi.org/10.1145/3414685.3417766

Christoph Gissler, Andreas Peer, Stefan Band, Jan Bender, and Matthias Teschner. 2019. Interlinked SPH Pressure Solvers for Strong Fluid-Rigid Coupling. *ACM Trans. Graph.* 38, 1 (Feb. 2019), 1–13. https://doi.org/10.1145/3284980

Eran Guendelman, Andrew Selle, Frank Losasso, and Ronald Fedkiw. 2005. Coupling Water and Smoke to Thin Deformable and Rigid Shells. In *ACM SIGGRAPH 2005 Papers* (Los Angeles, California) *(SIGGRAPH '05)*. Association for Computing Machinery, New York, NY, USA, 973–981. https://doi.org/10.1145/1186822.1073299

David Hahn, Pol Banzet, James M Bern, and Stelian Coros. 2019. Real2Sim: Visco-elastic parameter estimation from dynamic motion. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–13.

Xuchen Han, Theodore F. Gast, Qi Guo, Stephanie Wang, Chenfanfu Jiang, and Joseph Teran. 2019. A Hybrid Material Point Method for Frictional Contact with Diverse Materials. *Proc. ACM Comput. Graph. Interact. Tech.* 2, 2, Article 17 (jul 2019), 24 pages. https://doi.org/10.1145/3340258

Nikolaus Hansen. 2006. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation* (2006), 75–102.

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

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. 2020. DiffTaichi: Differentiable Programming for Physical Simulation. https://doi.org/10.48550/arXiv.1910.00935 arXiv:1910.00935 [physics, stat].

Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 2018a. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.* 37, 4 (Aug. 2018), 1–14. https://doi.org/10.1145/3197517.3201293

Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B. Tenenbaum, William T. Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. 2018b. ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics. *arXiv:1810.01054 [cs]* (Oct. 2018). http://arxiv.org/abs/1810.01054 arXiv: 1810.01054.

Markus Ihmsen, Nadir Akinci, Markus Becker, and Matthias Teschner. 2011. A Parallel SPH Implementation on Multi-Core CPUs. *Computer Graphics Forum* 30, 1 (2011), 99–112. https://doi.org/10.1111/j.1467-8659.2010.01832.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2010.01832.x

Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2014. Implicit Incompressible SPH. *IEEE Trans. Visual. Comput. Graphics* 20, 3 (March 2014), 426–435. https://doi.org/10.1109/TVCG.2013.105

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980

Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. 2016. Drucker-Prager Elastoplasticity for Sand Animation. *ACM Trans. Graph.* 35, 4, Article 103 (jul 2016), 12 pages. https://doi.org/10.1145/2897824.2925906

Bryan M. Klingner, Bryan E. Feldman, Nuttapong Chentanez, and James F. O'Brien. 2006. Fluid Animation with Dynamic Meshes. In *ACM SIGGRAPH 2006 Papers* (Boston, Massachusetts) *(SIGGRAPH '06)*. Association for Computing Machinery, New York, NY, USA, 820–825. https://doi.org/10.1145/1179352.1141961

Dan Koschier and Jan Bender. 2017. Density maps for improved SPH boundary handling. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '17)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3099564.3099565

D Koschier, J Bender, B Solenthaler, and M Teschner. 2019. SPH Techniques for the Physics Based Simulation of Fluids and Solids. (2019).

Yifei Li, Tao Du, Sangeetha Grama Srinivasan, Kui Wu, Bo Zhu, Eftychios Sifakis, and Wojciech Matusik. 2022. Fluidic Topology Optimization with an Anisotropic Mixture Model. *ACM Trans. Graph.* 41, 6 (Nov. 2022), 239:1–239:14. https://doi.org/10.1145/3550454.3555429

Yifei Li, Tao Du, Kui Wu, Jie Xu, and Wojciech Matusik. 2023. DiffCloth: Differentiable Cloth Simulation with Dry Frictional Contact. *ACM Trans. Graph.* 42, 1 (Feb. 2023), 1–20. https://doi.org/10.1145/3527660 arXiv:2106.05306 [cs].

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. 2018. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566* (2018).

Junbang Liang, Ming Lin, and Vladlen Koltun. 2019. Differentiable cloth simulation for inverse problems. *Advances in Neural Information Processing Systems* 32 (2019).

Ran Luo, Weiwei Xu, Tianjia Shao, Hongyi Xu, and Yin Yang. 2019. Accelerated complex-step finite difference for expedient deformable simulation. *ACM Trans. Graph.* 38, 6 (Dec. 2019), 1–16. https://doi.org/10.1145/3355089.3356493

Pingchuan Ma, Tao Du, John Z. Zhang, Kui Wu, Andrew Spielberg, Robert K. Katzschmann, and Wojciech Matusik. 2021. DiffAqua: a differentiable computational design pipeline for soft underwater swimmers with shape interpolation. *ACM Trans. Graph.* 40, 4 (Aug. 2021), 1–14. https://doi.org/10.1145/3450626.3459832

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 (Aug. 2018), 1–11. https://doi.org/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. https://doi.org/10.1145/1015706.1015744

J. J. Monaghan. 1992. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics* 30 (Jan. 1992), 543–574. https://doi.org/10.1146/annurev.aa.30.090192.002551 ADS Bibcode: 1992ARA&A..30..543M.

J Krishna Murthy, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, et al. 2020. gradSim: Differentiable simulation for system identification and visuomotor control. In *International Conference on Learning Representations*.

Shin-ichiro Nagahiro and Yoshinori Hayakawa. 2005. Theoretical and numerical approach to "magic angle" of stone skipping. *Physical review letters* 94, 17 (2005), 174501.

Rahul Narain, Abhinav Golas, and Ming C. Lin. 2010. Free-Flowing Granular Materials with Two-Way Solid Coupling. In *ACM SIGGRAPH Asia 2010 Papers* (Seoul, South Korea) *(SIGGRAPH ASIA '10)*. Association for Computing Machinery, New York, NY, USA, Article 173, 10 pages. https://doi.org/10.1145/1866158.1866195

Elvis Nava, John Z. Zhang, Mike Y. Michelis, Tao Du, Pingchuan Ma, Benjamin F. Grewe, Wojciech Matusik, and Robert K. Katzschmann. 2022. Fast Aquatic Swimmer Optimization with Differentiable Projective Dynamics and Neural Network Hydrodynamic Models. http://arxiv.org/abs/2204.12584 arXiv:2204.12584 [physics].

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. 2020. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409* (2020).

Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. 2021. Efficient Differentiable Simulation of Articulated Bodies. *ICML* (2021), 19.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics* 378 (2019), 686–707.

Brener Ramos, Felix Trost, and Nils Thuerey. 2022. Control of Two-way Coupled Fluid Systems with Differentiable Solvers. *arXiv preprint arXiv:2206.00342* (2022).

J. Rapin and O. Teytaud. 2018. Nevergrad - A gradient-free optimization platform. https://GitHub.com/FacebookResearch/Nevergrad.

Bo Ren, Xiaohan Ye, Zherong Pan, and Taiyuan Zhang. 2023. Versatile Control of Fluid-directed Solid Objects Using Multi-task Reinforcement Learning. *ACM Trans. Graph.* 42, 2 (April 2023), 1–14. https://doi.org/10.1145/3554731

Avi Robinson-Mosher, Tamar Shinar, Jon Gretarsson, Jonathan Su, and Ronald Fedkiw. 2008. Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Transactions on Graphics (TOG)* 27, 3 (2008), 1–9.

Connor Schenck and Dieter Fox. 2018. Spnets: Differentiable fluid dynamics for deep neural networks. In *Conference on Robot Learning*. PMLR, 317–335.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. https://doi.org/10.48550/ARXIV.1707.06347

Siyuan Shen, Yin Yang, Tianjia Shao, He Wang, Chenfanfu Jiang, Lei Lan, and Kun Zhou. 2021. High-order differentiable autoencoder for nonlinear model reduction. *ACM Trans. Graph.* 40, 4 (Aug. 2021), 1–15. https://doi.org/10.1145/3450626.3459754

Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective incompressible SPH. In *ACM SIGGRAPH 2009 papers*. 1–6.

Tetsuya Takahashi and Christopher Batty. 2020. Monolith: a monolithic pressure-viscosity-contact solver for strong two-way rigid-rigid rigid-fluid coupling. *ACM Trans. Graph.* 39, 6 (Dec. 2020), 1–16. https://doi.org/10.1145/3414685.3417798

Tetsuya Takahashi, Yoshinori Dobashi, Tomoyuki Nishita, and Ming C. Lin. 2018. An Efficient Hybrid Incompressible SPH Solver with Interface Handling for Boundary Conditions. *Computer Graphics Forum* 37, 1 (2018), 313–324. https://doi.org/10.1111/cgf.13292 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13292

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

Tetsuya Takahashi and Ming C. Lin. 2019. A Geometrically Consistent Viscous Fluid Solver with Two-Way Fluid-Solid Coupling. *Computer Graphics Forum* (2019). https://doi.org/10.1111/cgf.13618

Jie Tan, Yuting Gu, Greg Turk, and C. Karen Liu. 2011. Articulated swimming creatures. In *ACM SIGGRAPH 2011 papers (SIGGRAPH '11)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/1964921.1964953

Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C. Karen Liu. 2021. Fast and Feature-Complete Differentiable Physics for Articulated Rigid Bodies with Contact. http://arxiv.org/abs/2103.16021 arXiv:2103.16021 [cs, eess].

Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. 2021a. An End-to-End Differentiable Framework for Contact-Aware Robot Design. In *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation. https://doi.org/10.15607/RSS.2021.XVII.008

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. 2021b. Accelerated Policy Learning with Parallel Differentiable Simulation. In *International Conference on Learning Representations*.