UNIVERSITY OF CALIFORNIA

Los Angeles

A Contact Proxy Splitting Method for Lagrangian Solid-Fluid Coupling

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Computer Science

by

Tianyi Xie

2023

ABSTRACT OF THE THESIS

A Contact Proxy Splitting Method for Lagrangian Solid-Fluid Coupling

by

Tianyi Xie
Master of Science in Computer Science
University of California, Los Angeles, 2023
Professor Demetri Terzopoulos, Chair

Accurate and robust simulation of solid and fluid dynamics with their interactions is non-trivial in computer graphics. In this thesis, we present a robust and efficient method for simulating Lagrangian solid-fluid coupling based on a new operator splitting strategy. We use variational formulations to approximate fluid properties and solid-fluid interactions, and introduce a unified two-way coupling formulation for SPH fluids and FEM solids using interior point barrier-based frictional contact. We split the resulting optimization problem into a fluid phase and a solid-coupling phase using a novel time-splitting approach with augmented *contact proxies*, and propose efficient custom linear solvers. Our technique accounts for fluids interaction with nonlinear hyperelastic objects of different geometries and codimensions, while maintaining an algorithmically guaranteed non-penetrating criterion. Comprehensive benchmarks and experiments demonstrate the efficacy of our method.

The thesis of Tianyi Xie is approved.

Quanquan Gu

Chenfanfu Jiang

Demetri Terzopoulos, Committee Chair

University of California, Los Angeles

2023

To my family and friends.

TABLE OF CONTENTS

1 Introduction **1**

1.1 Thesis Contributions 2

1.2 Thesis Overview 3

2 Related Work **4**

2.1 Coupling Solids with Eulerian Fluids 4

2.2 Coupling Solids with Lagrangian Fluids 5

2.3 Hybrid Methods with Coupled Phases and Contact 6

3 Background **8**

3.1 Time Integration 8

3.2 Hyperelasticity 9

3.3 Smoothed Particle Hydrodynamics 11

4 Formulation **15**

4.1 Governing Equations 15

4.2 Solid Domain 16

4.3 Fluid Domain 16

4.3.1 Incompressibility Potential 17

4.3.2 Viscosity Potential 18

4.4 Coupling 19

4.4.1 Barrier Potential for Non-penetration 19

4.4.2 Friction Potential 21

4.5 Optimization Time Integrator 21

5 Efficient Solver **23**

5.1 Time Splitting 24

5.1.1 Baseline Time Splitting 24

5.1.2 Time Splitting with Contact Proxy 25

5.2 Solving Linear Systems 27

5.2.1 Fluid Phase 27

5.2.2 Solid-Coupling Phase 28

6 Experiments **30**

6.1 Ablation Study 30

6.1.1 Time Splitting Evaluation 30

6.1.2 Linear Solver Evaluation 34

6.2 Comparisons 36

6.2.1 Fluid Dynamics 36

6.2.2 Solid-Fluid Coupling 37

6.3 Complex Scenarios 38

7 Conclusions and Future Work **43**

A Derivatives of Fluid Potentials **45**

B Time Splitting **47**

B.1 Baseline Time Splitting 47

B.2 Error Analysis 48

References **50**

6.1 Timing breakdown of simulations . . . 31

6.2 Bob simulated with different time stepping schemes . . . 32

6.3 Viscous objects on cloth simulated with different time stepping schemes . . . 33

6.4 Statistics of simulations with different stiffness parameter . . . 34

6.5 Shoot armadillo . . . 35

6.6 Dam break . . . 36

6.7 Liquid Bunnies . . . 38

6.8 Buoyancy . . . 39

6.9 Varying friction . . . 39

6.10 Twist cylinder . . . 39

6.11 Cream . . . 40

6.12 Angry cow . . . 41

6.13 Kicking water . . . 41

LIST OF TABLES

6.1 Simulation statistics 31

6.2 Statistics of different time stepping schemes 32

6.3 Comparison of different linear system solvers 35

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to Professor Chenfanfu Jiang for his invaluable guidance and support throughout the research reported herein, as well as for providing me with the opportunity to pursue my PhD degree at UCLA. Without his mentorship and encouragement, I would not have been able to accomplish this significant work. I am also grateful to Professor Demetri Terzopoulos for his guidance on this thesis and for serving as my co-advisor. I also thank Professor Quanquan Gu for serving as my thesis committee member.

I express my appreciation to Professor Minchen Li for mentoring me during this project. I thank him for sharing his knowledge of physics-based simulation and numerical optimization when I began to work on this research project. Minchen helped me develop the necessary skills to conduct graphics research.

I thank my lab mates at the UCLA MultiPLES Lab: Yu Fang, Ziyin Qu, Xuan Li, Yunuo Chen, Yuxing Qiu, Yadi Cao, Zeshun Zong, and Ying Jiang. Their camaraderie, advice, and support were invaluable to me.

Lastly, I am deeply grateful to my family and friends for their unwavering support and encouragement. I am particularly grateful to my parents for their unconditional love and support throughout my academic journey, which have been instrumental in helping me achieve my goals.

VITA

2020	B.S., Software Engineering Shanghai Jiao Tong University
2023	M.S. Candidate, Computer Science University of California, Los Angeles
2020-2021	Research Intern Bytedance AI Lab

CHAPTER 1

Introduction

The coupling of solids and fluids is common in nature but challenging to simulate. While solids are typically simulated using Lagrangian meshes, fluids are often discretized using Eulerian grids to accommodate topology changes. To accurately couple these distinct discretizations, sophisticated algorithms, such as the cut-cell method (Zarifi and Batty, 2017), are often necessary. Unfortunately, they can be computationally expensive and do not handle thin shells. Purely Eulerian schemes (Teng et al., 2016; Valkov et al., 2015) or Smoothed-Particle Hydrodynamics (SPH) (Akinci et al., 2012; Gissler et al., 2019) have demonstrated successful two-way coupling by same-view discretization, but they do not easily extend to nonlinear elastodynamics. Hybrid methods like the Material Point Method (MPM) (Jiang et al., 2016) can simulate mixed materials, but can experience artificial stickiness unless resolved with more expensive schemes (Fang et al., 2020). Furthermore, these methods do not ensure non-intersecting trajectories and often require additional correction procedures to handle accidentally penetrated fluids during advection.

We take the Lagrangian approach and present a new method for coupling Finite Element Method (FEM) solids and SPH fluids. By approximating solid, fluid, and interaction terms with potentials, we formulate two-way coupling as a monolithic optimization problem. Specifically, we draw inspiration from position-based fluids (Macklin and Müller, 2013) and model weak incompressibility using a quadratic energy and a new *updated Lagrangian update rule* to track volume changes. We further symmetrize the discrete Laplacian-based viscosity and propose a *discrete quadratic potential* for better accuracy and robustness. We follow the Incremental Potential Contact (IPC) (Li et al., 2020) model to enforce guaranteed separable boundary conditions and resolve frictional contacts at the interface.

The proposed formulation achieves strong coupling, but can be exceedingly inefficient when solved with Newton's method due to the huge and dense Hessian of the fluid component, which is a result of the need for many particle neighbors to achieve an accurate SPH discretization. This causes a significant computational bottleneck.

To tackle this issue, we propose a robust *proxy contact energy* formulation, splitting the time integration into a fluid phase and a solid-coupling phase. The fluid phase requires only one Newton iteration per time step, resulting in increased efficiency with nonlinear optimization occurring only during the solid-coupling phase. One of the key advantages of our quadratic proxy is its ability to effectively resolve instability caused by time splitting. This is achieved through its asymptotic approximation to the solid-fluid contact force. Additionally, the consistency of the time integration is maintained through the cancellation of the proxy's contribution in the solid-coupling phase, resulting in only a small splitting error. Finally, we design a matrix-free conjugate gradient solver and a domain-decomposed solver to further enhance the computational efficiency.

1.1 Thesis Contributions

The contributions of this thesis include

- a unified penetration-free two-way coupling framework for weakly compressible SPH fluids and nonlinear elastic FEM solids in arbitrary codimensions;
- consistently modeled incompressibility and viscosity potentials for SPH fluids, incorporating advantages of Updated Lagrangian (UL) kinematics;
- a robust time splitting scheme with contact proxies that enables separate time integrations of solids and fluids;
- a matrix-free Conjugate Gradient (CG) solver and a Schur-complement-based domain-decomposed solver for efficiently solving the linear systems.

1.2 Thesis Overview

The remainder of this thesis is organized as follows:

- Chapter 2 surveys previous research on simulating solid-fluid coupling.
- Chapter 3 introduces background theory, including time integration, hyperelasticity, and SPH.
- Chapter 4 presents the mathematical formulation of our solid-fluid coupling system, from a continuous form to a discrete one. Furthermore, the resulting time integration is converted into an unconstrained optimization problem.
- Chapter 5 describes our proposed proxy-augmented time splitting scheme with custom linear system solvers.
- Chapter 6 evaluates our methods across various benchmarks and experiments.
- Chapter 7 concludes by reviewing the key contributions of this thesis and discusses promising topics for future work.

CHAPTER 2

Related Work

2.1 Coupling Solids with Eulerian Fluids

Traditional fluid solvers typically use grid discretization to simulate fluid dynamics—aka Eulerian Fluids. Since the Lagrangian view dominates solid simulation, much existing literature focuses on coupling Eulerian fluids with Lagrangian solids by resolving interactions between the grids and irregular mesh boundaries. The ghost fluid method (Fedkiw et al., 1999; Fedkiw, 2002) was proposed to additionally discretize the Eulerian/Lagrangian interface. Early efforts considered weak coupling (Guendelman et al., 2005), which advances the solids and fluids in an alternating manner. On the other hand, strong coupling (Klingner et al., 2006) solves a monolithic system and is often more robust. The cut-cell method (Roble et al., 2005) is another widely used solution to accurately model the interactions between Lagrangian solids and Eulerian fluids, often through the usage of virtual nodes. This method is first introduced to computer graphics by Roble et al. (2005). Batty et al. (2007) proposed a variational framework to strongly couple fluids and rigid bodies by casting the pressure solve as a minimization problem. Subsequent extensions support deformable objects and thin shells (Robinson-Mosher et al., 2008, 2011), where the elastic forces are explicitly applied and the solid damping was solved together with fluid dynamics in an implicit manner. Assuming co-rotated linear elasticity, Zarifi and Batty (2017) incorporated the implicit solid dynamics into pressure projection, obtaining a symmetric positive-definite system. Later efforts also explored rigid-rigid (Takahashi and Batty, 2020) and rigid-fluid (Takahashi and Batty, 2021) frictional contacts.

Apart from Lagrangian solids, also explored was Eulerian discretization (Levin et al., 2011), where coupling can be conveniently achieved in a purely Eulerian fashion (Teng et al., 2016; Valkov et al., 2015). However, such discretization entails increased numerical dissipation, resulting in artifacts such as volume loss, and the inability to handle structures thinner than a grid cell. More recently, Brandt et al. (2019) built upon the immersed boundary method (Peskin, 2002) and proposed a reduced solver to simulate real-time coupling, focusing on incompressible elastic materials and no-slip boundary conditions.

2.2 Coupling Solids with Lagrangian Fluids

Fluids can also be directly modeled with Lagrangian meshes (Clausen et al., 2013; Klingner et al., 2006; Wang et al., 2020; Batty et al., 2012), enabling explicit coupling with solids. However, this inevitably incurs frequent remeshing operations, thus degrading the overall performance.

Another strategy is to use particle-based methods, or meshless methods. SPH (Koschier et al., 2022) is a meshless approximation strategy based on spatial sampling for continuous functions, which has shown its ability to produce visually compelling results in physics-based simulation, especially for the fluid. Pioneering researchers (Monaghan, 1994; Becker and Teschner, 2007) used the Equation of State (EOS) for weakly compressible fluids, where the pressure is proportional to the density deviation. However, this explicit formulation may strictly restrict time step sizes resulting in limited performance. Incompressibility has also been enforced by solving a Pressure Poisson Equation (PPE) (Solenthaler and Pajarola, 2009; Ihmsen et al., 2013; Bender and Koschier, 2015). This approach seeks to cancel out density or velocity divergence deviations caused by non-pressure forces through the use of pressure accelerations. SPH boundary handling techniques have been developed to prevent penetrations of fluid particles near solid boundaries (Becker et al., 2009b; Ihmsen et al., 2010; Becker and Teschner, 2007). One such method, proposed by Akinci et al. (2012), uses a single layer of boundary samples and has been applied to the coupling of fluids with both rigid bodies and elastic solids (Akinci et al., 2013). Gissler et al. (2019) proposed a global formulation that unifies rigid body and fluid dynamics, in which the fluid pressure solver is linked to a second artificial pressure solver for rigid body particles. Koschier and Bender (2017) introduced an alternative method using density maps to represent dynamic rigid boundaries, eliminating the need for boundary particles. Bender et al. (2019) proposed using the volume contribution of boundary geometry to compute boundary forces, which reduces the cost of precomputation. However, this type of method cannot be applied to deformable bodies due to the expensive cost of computing maps at run time.

Solenthaler et al. (2007) used SPH to approximate the deformation gradient of linear elastic materials, but the resulting gradient is not rotation invariant. Becker et al. (2009a) addressed this issue by using shape matching to determine orientation and calculating forces in a rotated configuration. Peer et al. (2018) proposed an implicit scheme and applied kernel gradient correction (Bonet and Lok, 1999) to obtain a first-order consistent SPH formulation for the deformation gradient. By integrating the solid particles into the preexisting fluid pressure solver, contact handling can be inherently resolved, but SPH still faces numerical issues, such as the zero-mode (Kugelstadt et al., 2021; Ganzenmüller, 2015) when simulating elastic objects. Additionally, the pressure solver will treat solid objects as incompressible under compression, which may not be applicable in all cases.

2.3 Hybrid Methods with Coupled Phases and Contact

The MPM (Sulsky et al., 1995; Jiang et al., 2016) is a hybrid Eulerian-Lagrangian approach that treats materials as a collection of material points. Stomakhin et al. (2013) first introduced this method into computer graphics for snow simulation. The MPM has also been proven successful in capturing solid-fluid coupling (Stomakhin et al., 2014; Fei et al., 2018; Yan et al., 2018) and mixtures (Tampubolon et al., 2017; Gao et al., 2018). While one advantage of the MPM is the automatic handling of contacts between different materials because of its hybrid nature, the separation of continuum and free-slip interaction among different objects remains challenging. This drawback can be attributed to the fact that the traditional MPM (Jiang et al., 2016) does not inherently support discontinuous velocities, resulting in sticky boundary conditions. To allow easy material separation, Hu et al. (2018) introduced a Compatible Particle-In-Cell (CPIC) algorithm to allow discontinuous velocities. Fang et al. (2020) proposed a free-slip treatment, but did not consider separation. Recently, a FEM-MPM coupling method based on a variational barrier formulation (Li et al., 2020) has been proposed for coupling frictional and separable elastic materials (Li et al., 2022). Our approach for solid-fluid coupling is inspired by this method and uses a similar purely Lagrangian framework.

CHAPTER 3

Background

3.1 Time Integration

The simulation of various materials can be performed by time integration, which usually takes a sequence of discrete time steps, considers the internal and external forces on objects, and integrates them over time to derive the velocity and position updates.

Taking as an example the most popular time integration scheme, the implicit Euler method, we stack the velocity and position of every simulated material particle at time step n into the vectors $\mathbf{v}^n$ and $\mathbf{x}^n$, respectively, and the method proceeds as

$$\begin{cases} \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \\ \mathbf{v}^{n+1} = \mathbf{v}^n + h\mathbf{M}^{-1}\left(\mathbf{f}_{\text{ext}} + \mathbf{f}_{\text{int}}(\mathbf{x}^{n+1})\right), \end{cases} \tag{3.1}$$

where h is the time step size, $\mathbf{M}$ is the mass matrix, $\mathbf{f}_{\text{ext}}$ is the external force (gravity), and $\mathbf{f}_{\text{int}}(\mathbf{x}^{n+1})$ is the internal force evaluated at $\mathbf{x}^{n+1}$. Solving this time integration equation yields the velocity and position vectors at the next time step $n+1$. An important feature of the implicit Euler method is unconditional stability, which means the integration result will never explode like explicit time integration.

The above implicit equation can usually be reformulated as an equivalent minimization problem:

$$\begin{aligned} \mathbf{x}^{n+1} &= \arg\min_{\mathbf{x}} E(\mathbf{x}), \\ E(\mathbf{x}) &= \frac{1}{2}\|\mathbf{x} - \tilde{\mathbf{x}}^n\|_{\mathbf{M}} + h^2 P(\mathbf{x}), \end{aligned} \tag{3.2}$$

where $\tilde{\mathbf{x}}^n = \mathbf{x}^n + h\mathbf{v}^n$ and $\frac{1}{2}\|\mathbf{x} - \tilde{\mathbf{x}}^n\|_{\mathbf{M}}$ is the inertial term, potential energy $P(\mathbf{x})$ accounts for both external and internal forces with $\frac{\partial P(\mathbf{x})}{\partial \mathbf{x}} = -(\mathbf{f}_{\text{ext}} + \mathbf{f}_{\text{int}}(\mathbf{x}))$, and $E(\mathbf{x})$ is called the incremental potential. This minimization problem is equivalent to the implicit equation of (3.1), since the first-order necessary condition $\nabla E(\mathbf{x}) = 0$ is satisfied at any minimal point. A standard way to solve this minimization problem is to apply a Newton-type method with line search. Specifically, at iteration i the optimization process proceeds as

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha \mathbf{H}(\mathbf{x}^i)^{-1} \nabla E(\mathbf{x}^i), \tag{3.3}$$

where $\mathbf{H}(\mathbf{x}^i)$ is a Symmetric Positive Definite (SPD) matrix containing second-order information at $\mathbf{x}^i$ and α is the step size calculated by line search, which ensures the overall potential is always decreasing.

3.2 Hyperelasticity

Hyperelasticity describes the ability of objects to retain their rest shape under deformation. To provide a concise mathematical description of deformation that an elastic object has sustained, a tensor field called the deformation gradient is defined as

$$\mathbf{F}(\mathbf{X}) = \frac{\partial \phi(\mathbf{X})}{\partial \mathbf{X}}, \tag{3.4}$$

where $\mathbf{X}$ is the position in material space Ω^0 and $\phi(\mathbf{X})$ is the deformation map that maps from the position in material space Ω^0 to that in world space Ω^t. For tetrahedral meshes, as the deformation map is typically taken to be piecewise linear across tetrahedra, the deformation gradient is then piecewise constant, which can be easily computed (in 3D) as

$$\mathbf{F} = \Big[\mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_3 - \mathbf{x}_1, \mathbf{x}_4 - \mathbf{x}_1\Big] \Big[\mathbf{X}_2 - \mathbf{X}_1, \mathbf{X}_3 - \mathbf{X}_1, \mathbf{X}_4 - \mathbf{X}_1\Big]^{-1}, \tag{3.5}$$

where $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$ are positions of the four nodes of a tetrahedron in world space, and $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$, and $\mathbf{X}_4$ are their positions in material space. Intuitively, the deformation gradient measures the amount of deformation in a local region.

We can make use of this local information to define constitutive models, i.e., the elastic energies that penalize deformation. A commonly used constitutive model for hyperelasticity is Neo-Hookean, whose energy density function is

$$\psi(\mathbf{F}) = \frac{\mu}{2}(\mathrm{tr}(\mathbf{F}^T\mathbf{F}) - 3) - \mu \ln J + \frac{\lambda}{2}(\ln J)^2, \tag{3.6}$$

where J is the volume ratio, which may be calculated as the determinant of $\mathbf{F}$, and μ and λ are lame parameters, which can be computed using the Young's modulus E and Poisson's ratio ν:

$$\mu = \frac{E}{2(1+\nu)}, \qquad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}. \tag{3.7}$$

E determines the stiffness of the material and ν its ability to preserve volume under deformation. Then the overall elastic energy can be obtained by integrating the energy density over the mesh domain

$$\Psi(\mathbf{X}) = \sum_t V_t \psi(\mathbf{X}), \tag{3.8}$$

where V_t is the volume of tetrahedron t. With this definition of elastic potential energy, the elasticity force can be derived by taking the derivative

$$\mathbf{f} = -\frac{\partial \Psi(\mathbf{X})}{\partial \mathbf{X}}. \tag{3.9}$$

One thing worth noting about the Neo-Hookean model is that $J > 0$ must always be satisfied during the simulation, otherwise the $\ln J$ term in (3.6) will be undefined. This means that mesh degeneration or inversion is not permitted. To avoid this issue, we can employ line search filtering to prevent too large a step size such that J remains positive for each tetrahedron during the optimization process.

3.3 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is a spatial discretization technique, which is commonly used for the numerical simulation of continuum mechanical problems. Recent works have demonstrated its ability to efficiently simulate various materials, especially fluids.

Foundation: The foundation of SPH begins with the Dirac function, which is defined as

$$\delta(\mathbf{r}) = \begin{cases} \infty, & \text{if } \mathbf{r} = 0, \\ 0, & \text{otherwise,} \end{cases} \tag{3.10}$$

and also satisfies the identity property $\int \delta(\mathbf{r}) dV = 1$. This can be understood as the limit of the Gaussian normal distribution as the variance approaches to zero. For any given continuous compactly supported function $A(\mathbf{x})$, its convolution with the Dirac distribution is identical to $A(\mathbf{x})$ itself. Formally,

$$A(\mathbf{x}) = (A * \delta)(\mathbf{x}) = \int A(\mathbf{x}')\delta(\mathbf{x} - \mathbf{x}')dV'. \tag{3.11}$$

It is hard to discretize this integration due to the non-smooth Dirac function. We can instead make a continuous approximation, which we refer to as the kernel function $W(\mathbf{r}, \hbar)$. Integral (3.11) is then approximated as

$$A(\mathbf{x}) \approx (A * W)(\mathbf{x}) = \int A(\mathbf{x}')W(\mathbf{x} - \mathbf{x}', \hbar)dV', \tag{3.12}$$

where $\hbar$ denotes the kernel's smoothing length, which controls how strongly the resulting value is affected by values in its close proximity. In the SPH literature, commonly used kernel functions include the cubic spline and spiky functions, among others. In a domain with sampling points (SPH particles), (3.12) can be discretized as the weighted summation over the sampling points

$$\begin{aligned}(A * W)(\mathbf{x}) &= \int \frac{A(\mathbf{x}')}{\rho(\mathbf{x}')} W(\mathbf{x}-\mathbf{x}', \hbar)\rho(\mathbf{x}')dV', \\ &\approx \sum_{j\in\mathcal{F}} A_j \frac{m_j}{\rho_j} W(\mathbf{x}-\mathbf{x}_j, \hbar),\end{aligned} \tag{3.13}$$

where $\mathcal{F}$ is the set of sampling points within the support region of the kernel function, A_j denotes the value evaluated at the respective position, i.e., $A_j = A(\mathbf{x}_j)$, and where m_j and ρ_j are, respectively, the mass and density of sampling point j. A nice and interesting feature of this formulation is that the particles are not required to carry with the density field. Specifically, taking $A(\mathbf{x})$ as $\rho(\mathbf{x})$, by (3.13), we can easily reconstruct the density field as

$$\rho_i = \sum_j m_j W_{ij}, \tag{3.14}$$

where W_{ij} is the abbreviation of $W(\mathbf{x}_i - \mathbf{x}_j, \hbar)$. Alternatively, the density can be tracked by numerical integration of the continuity equation, which describes the density evolution $\frac{d\rho}{dt} = -\rho\nabla\cdot\mathbf{v}$. However, this approach is less robust due to the accumulation error of numerical integration.

Discretization of Differential Operators: Besides the discretization of field quantities, it is usually necessary to discretize spatial differential operators in order to numerically solve physical conservation laws. A straightforward way is to directly take the derivative of (3.13), but the resulting approximation is not first-order accurate. To recover the accuracy, a difference formula is introduced to estimate the gradient quantities. Formally,

$$\nabla A_i = \sum_j \frac{m_j}{\rho_j}(A_j - A_i)\nabla_i W_{ij}. \tag{3.15}$$

The same formula can also be straightforwardly applied to higher-dimensional functions. Another discrete formula for the gradient estimation can be derived, called the symmetric formula:

$$\nabla A_i = \rho_i \sum_j m_j \left(\frac{A_i}{\rho_i^2} + \frac{A_j}{\rho_j^2} \right) \nabla_i W_{ij}. \tag{3.16}$$

The main advantage of the symmetric formula is that the discrete forces estimated by it exactly conserve linear and angular momentum, which is essential in certain simulations. But note that the symmetric formula is not first-order accurate.

Similarly, a improved Laplacian estimation is proposed by Brookshaw (1985):

$$\nabla^2 A_i = -\sum_i \frac{m_j}{\rho_j} A_{ij} \frac{2\| \nabla_i W_{ij}\|}{\|\mathbf{x}_{ij}\|}, \tag{3.17}$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. The main idea of this formulation is to use only the first-order derivative of the kernel function and the second-order derivative is approximated using a finite-difference-like operation. The second-order derivatives of vectorial field quantities are realized analogously:

$$\nabla^2 \mathbf{A}_i = -\sum_i \frac{m_j}{\rho_j} \mathbf{A}_{ij} \frac{2\| \nabla_i W_{ij}\|}{\|\mathbf{x}_{ij}\|}, \tag{3.18}$$

$$\nabla(\nabla \cdot \mathbf{A}_i) = \sum_j \frac{m_j}{\rho_j} [(d+2)(\mathbf{A}_{ij} \cdot \tilde{\mathbf{x}}_{ij})\tilde{\mathbf{x}}_{ij} - \mathbf{A}_{ij}] \frac{\|\nabla_i W_{ij}\|}{\|\mathbf{x}_{ij}\|}, \tag{3.19}$$

where d denotes the spatial dimension and $\tilde{\mathbf{x}}_{ij} = \frac{\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|}$ is the normalized distance vector. Still, the physical forces estimated by the Laplacian operator in (3.18) is not momentum conserving. This issue can be resolved by combining (3.18) and (3.19), yielding

$$\sum_j \frac{m_j}{\rho_j} (\mathbf{A}_{ij} \cdot \tilde{\mathbf{x}}_{ij})\tilde{\mathbf{x}}_{ij} \frac{\|\nabla_i W_{ij}\|}{\|\mathbf{x}_{ij}\|} = \frac{\nabla(\nabla \cdot \mathbf{A}_i)}{d+2} - \frac{\nabla^2 \mathbf{A}_i}{2(d+2)}. \tag{3.20}$$

In the context of divergence-free fields, the Laplacian operator can then be discretized as

$$\nabla^2 \mathbf{A}_i = 2(d+2) \sum_j \frac{m_j}{\rho_j} (\mathbf{A}_{ij} \cdot \tilde{\mathbf{x}}_{ij})\tilde{\mathbf{x}}_{ij} \frac{\|\nabla_i W_{ij}\|}{\|\mathbf{x}_{ij}\|}. \tag{3.21}$$

Thus, the resulting forces of the Laplacian operator act only along the line between two interacting particles i and j.

CHAPTER 4

Formulation

In this chapter, we derive a time integrator for a coupled system of solids and fluids by starting with the governing equations and then performing discretization. Subscripts s and f will represent solid and fluid quantities, respectively.

4.1 Governing Equations

The governing equations for the coupled system are

$$\rho_s \frac{D\mathbf{v}_s}{Dt} = \nabla \cdot \sigma + \rho_s \mathbf{g} + \mathbf{f}_{s\to s} + \mathbf{f}_{f\to s}, \tag{4.1}$$

$$\rho_f \frac{D\mathbf{v}_f}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{v}_f + \rho_f \mathbf{g} - \mathbf{f}_{f\to s}, \tag{4.2}$$

$$\nabla \cdot \mathbf{v}_f = 0, \tag{4.3}$$

where ρ is density, $\mathbf{g}$ is gravity, $\mathbf{f}_{s\to s}$ is the self-contact force of solids, $\mathbf{f}_{f\to s}$ is the contact force exerted by fluids, σ is Cauchy stress, p is pressure, and μ is the dynamic viscosity (Bridson, 2015).

At the interface between solids and fluids, we enforce the separable boundary condition

$$0 \leq (\mathbf{v}_s - \mathbf{v}_f) \cdot \mathbf{n}_f \perp (\mathbf{f}_{f\to s} \cdot \mathbf{n}_f) \geq 0 \tag{4.4}$$

to prevent penetration while allowing separation (Batty et al., 2007). This condition helps determine the normal component of $\mathbf{f}_{f\to s}$. For the tangential component (friction), let $\mathbf{u} = (\mathbf{I} - \mathbf{n}_f \otimes \mathbf{n}_f)(\mathbf{v}_s - \mathbf{v}_f)$ be the tangential relative velocity. Following the Maximum

Dissipation Principle (Moreau, 2011), we then have

$$\begin{aligned} (\mathbf{I} - \mathbf{n}_f \otimes \mathbf{n}_f)\mathbf{f}_{f\to s} &= \arg\min_{\boldsymbol{\beta}} \boldsymbol{\beta} \cdot \mathbf{u}, \\ \text{s.t.} \quad \|\boldsymbol{\beta}\| &\le \mu_t \mathbf{f}_{f\to s} \cdot \mathbf{n}_f \;\; \text{and} \;\; \boldsymbol{\beta} \cdot \mathbf{n}_f = 0, \end{aligned} \tag{4.5}$$

where μ_t is the friction coefficient. We enforce exact mass conservation by adopting Lagrangian methods to discretize both domains.

4.2 Solid Domain

We focus on nonlinear hyperelastic solids, where the elastic force is the negative gradient of an elastic potential. After discretizing the solid domain Ω_s into Lagrangian linear finite elements (triangles in 2D and tetrahedra in 3D), the total elastic potential is a piecewise constant summation of an elastic energy density function $\psi_s(\mathbf{F})$ (e.g., neo-Hookean) over the mesh domain:

$$\Psi_s(\mathbf{x}) = \sum_e V_e \psi_s(\mathbf{F}_e), \tag{4.6}$$

where V_e is the rest volume of tetrahedron e, and $\mathbf{F} = \frac{\partial \mathbf{x}(\mathbf{X},t)}{\partial \mathbf{X}}$ is the deformation gradient with $\mathbf{X}$ and $\mathbf{x}$ denoting the material and world space coordinates, respectively (Sifakis and Barbic, 2012). For $\mathbf{f}_{s\to s}$, we follow the smooth barrier approach of Li et al. (2020) that guarantees non-penetration. We leave the discussion of $\mathbf{f}_{f\to s}$ to Section 4.4.

4.3 Fluid Domain

Following the SPH literature (Becker and Teschner, 2007; Macklin and Müller, 2013; Ihmsen et al., 2013; Bender and Koschier, 2015), we discretize the fluid domain Ω_f using Lagrangian particles. To simulate fluids using optimization-based time integration, we approximate both the pressure and viscosity forces as conservative forces. We verify that these two proposed potential energies are both convex and quadratic in Appendix A.

4.3.1 Incompressibility Potential

Pressure forces help preserve the volume of incompressible fluids. We thus model the incompressibility via a quadratic energy density function

$$\psi_{f,I}(J) = \frac{k_I}{2}(J-1)^2, \tag{4.7}$$

which penalizes the deviation from 1.0 of the volume ratio $J = \rho_0/\rho$, where ρ_0 is the initial density. The use of a large stiffness k_I in a convergent solver results in negligible visual compression, eliminating the need for higher degree polynomials in nearly incompressible fluids (Hyde et al., 2020). The incompressibility potential is obtained by integrating $\psi_{f,I}(J)$ over the fluid domain Ω_f^0 in *material space*:

$$P_I(\mathbf{x}) = \sum_i \frac{k_I}{2} V_0 (J_i(\mathbf{x}) - 1)^2, \tag{4.8}$$

where we assume that all fluid particles have equal rest volume V_0 and where J_i denotes the volume ratio of particle i as a function of $\mathbf{x}$.

Updated Lagrangian: The SPH literature often relates ρ to $\mathbf{x}$ through density summation in the world space. To obtain a linear relation between J and $\mathbf{x}$ so that the incompressibility potential stays quadratic in terms of $\mathbf{x}$, we track J in an *updated Lagrangian* fashion. Treating Ω^n as an intermediate reference space and differentiating the deformation map between Ω^n and Ω^{n+1} results in an update rule

$$J_i^{n+1} = J_i^n (1 + h \nabla \cdot \mathbf{v}_i^{n+1}), \tag{4.9}$$

the terms of which can be approximated via SPH as

$$J_i^n = \frac{\rho_0}{\sum\limits_j m_j W_{ij}}, \tag{4.10}$$

$$\nabla \cdot \mathbf{v}_i^{n+1} = \sum_j \frac{m_j}{\rho_j^n}(\mathbf{v}_j^{n+1} - \mathbf{v}_i^{n+1}) \cdot \nabla_i W_{ij}, \tag{4.11}$$

and $W_{ij} = W(\mathbf{x}_i - \mathbf{x}_j)$ is a kernel function (e.g., Cubic Spline kernel (Monaghan, 1992, 2005) or Spiky kernel (Müller et al., 2003)). Here J_i^n denote the reinitialized volume ratio of fluid particle i at the start of time step n. Such reinitialization avoids accumulated density and particle distribution errors commonly seen in other updated Lagrangian solvers like MPM.

4.3.2 Viscosity Potential

The viscosity of fluids is a measure of their resistance to shearing at a given rate, and it plays a vital role in producing effects like buckling and coiling. It is possible to model viscosity via strain rate tensors (Peer et al., 2015; Peer and Teschner, 2016; Takahashi et al., 2015; Bender and Koschier, 2016), but this may suffer from artifacts at the surface due to particle deficiencies in the SPH approximation. Following Monaghan (2005), we use the more robust velocity Laplacian (Weiler et al., 2018) and derive its energy form.

Combining SPH first-order derivatives and finite differences, the viscosity force can be computed as

$$\begin{aligned}\mathbf{f}_i(\mathbf{x}) &= \nu m_i \nabla^2 \mathbf{v}_i^{n+1} \\ &= 2\nu(d+2)\sum_j \frac{m_i m_j}{\rho_j} \frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2 + 0.01\hbar^2}\mathbf{v}_{ij}^{n+1},\end{aligned} \tag{4.12}$$

where $\mathbf{x}_{ij}^n = \mathbf{x}_i^n - \mathbf{x}_j^n$ and $\mathbf{v}_{ij}^{n+1} = \mathbf{v}_i^{n+1} - \mathbf{v}_j^{n+1}$, and where $\hbar$ is the support radius of the kernel, ν is the kinematic viscosity, and $d \in \{2, 3\}$ denotes the spatial dimensionality. Directly applying this force violates momentum conservation as the mutual interaction forces are unequal. To solve this problem, we perform a further approximation:

$$\mathbf{f}_i(\mathbf{x}) \approx 4\nu(d+2)\sum_j \frac{m_i m_j}{\rho_i+\rho_j}\frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2+0.01\hbar^2}\mathbf{v}_{ij}^{n+1}, \tag{4.13}$$

and also make the force integrable. Letting $\mathbf{V}_{ij} = 4(d+2)\frac{m_i m_j}{\rho_i+\rho_j}\frac{(-\nabla_i W_{ij})(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2+0.01\hbar^2}$, we can gather and integrate all the viscosity forces and obtain a quadratic viscosity potential

$$P_V(\mathbf{x}) = \frac{1}{4}\nu\hat{h}\sum_i\sum_j \|\mathbf{v}_{ij}^{n+1}\|^2_{\mathbf{V}_{ij}}, \tag{4.14}$$

where $\hat{h}$ is a constant scalar related to the time integration scheme. For example, $\hat{h} = h$ for the implicit Euler method as $\mathbf{v}^{n+1} = (\mathbf{x}^{n+1} - \mathbf{x}^n)/h$.

4.4 Coupling

4.4.1 Barrier Potential for Non-penetration

To couple the solid domain Ω_s with the fluid domain Ω_f, we use the separable boundary condition (4.4), which enforces non-interpenetration constraints between these two domains. To model these constraints, we first define a function

$$d(\partial\Omega_s^t, \mathbf{x}_f) = \min_{\mathbf{x}_s} \|\mathbf{x}_s - \mathbf{x}_f\|, \quad \mathbf{x}_s \in \partial\Omega_s^t,\ \mathbf{x}_f \in \Omega_f^t, \tag{4.15}$$

which measures the distance between $\mathbf{x}_f$, a point in the fluid domain, and the surface of the solid domain. Then the primal component of the constraints can be expressed as

$$d(\partial\Omega_s^t, \mathbf{x}_f) \geq 0, \quad \forall t \geq 0,\ \forall \mathbf{x}_f \in \Omega_f^t. \tag{4.16}$$

We then adopt the barrier formulation from Li et al. (2020) to model all the constraints in (4.4) between solids and fluids, and obtain a barrier potential

$$\int_{\partial\Omega_f^t} b(d(\partial\Omega_s^t, \mathbf{x}_f), \hat{d}) d\mathbf{x}_f, \tag{4.17}$$

where the barrier energy density $b(d, \hat{d})$ is piecewise smooth and is only activated when $d < \hat{d}$, improving efficiency and approximately satisfying the complimentarity slackness condition. As d approaches 0, the value of $b(d, \hat{d})$ monotonically increases to infinity, providing arbitrarily large repulsion to avoid interpenetration.

Since our solid and fluid domains are respectively discretized as meshes and particles, the barrier potential (4.17) in 3D can be numerically integrated as

$$\begin{aligned} B_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) &= \sum_{q \in \mathcal{Q}_f} s_q b(\min_{e \in \mathcal{B}_s} d^{PT}(\mathbf{x}_q, e), \hat{d}) \\ &= \sum_{q \in \mathcal{Q}_f} s_q \max_{e \in \mathcal{B}_s} b(d^{PT}(\mathbf{x}_q, e), \hat{d}), \end{aligned} \tag{4.18}$$

where $\mathcal{Q}_f$ is the set of all SPH fluid particles, $\mathcal{B}_s$ is the set of all the boundary triangles of the solids, $s_q = \pi(\frac{3J_nV_0}{4\pi})^{\frac{2}{3}}$ is the integration weight (boundary area) of each fluid particle, and $d_{PT}(\mathbf{x}_q, e)$ measures the distance between particle $\mathbf{x}_q$ and triangle e. Here, the min-max transformation is based on the non-ascending property of the barrier function. However, the max operator makes the barrier potential challenging to be efficiently optimized by gradient-based methods. Fortunately, due to the local support of the barrier function $b(d, \hat{d})$ as $\hat{d}$ is small, we can simply approximate the barrier potential as

$$B_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) = \sum_{q \in \mathcal{Q}_f} \sum_{e \in \mathcal{B}_s} s_q b(d^{PT}(\mathbf{x}_q, e), \hat{d}), \tag{4.19}$$

which may result in overestimated contact forces near the edges and nodes on the mesh boundary, but we have not observed any artifacts in our experiments.

4.4.2 Friction Potential

Following Li et al. (2020), we model the local friction forces $\mathbf{f}_k$ for every active solid-fluid contact pair k. Formally, the friction force is defined as

$$\mathbf{f}_k(\mathbf{x}_s, \mathbf{x}_f) = -\mu_t \lambda_k T_k(\mathbf{x}_s, \mathbf{x}_f) f_1(\|\mathbf{u}_k\|) \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}, \tag{4.20}$$

where λ_k is the contact force magnitude, $T_k(\mathbf{x}_s, \mathbf{x}_f) \in \mathbb{R}^{3n \times 2}$ is the consistently oriented sliding basis, and $\mathbf{u}_k$ is the relative sliding displacement, which can be computed as $\mathbf{u}_k = T_k(\mathbf{x}_s, \mathbf{x}_f)^T([\mathbf{x}_s^T, \mathbf{x}_f^T]^T - [(\mathbf{x}_s^n)^T, (\mathbf{x}_f^n)^T]^T)$. Here, f_1 is a smoothly approximated function designed for a smooth transition between the sticking and sliding modes. To make this friction formulation fit into optimization time integration, Li et al. (2020) further approximated the sliding basis $T(\mathbf{x}_s, \mathbf{x}_f)$ and contact force $\lambda_k(\mathbf{x}_s, \mathbf{x}_f)$ explicitly as $T(\mathbf{x}_s^n, \mathbf{x}_f^n)$ and $\lambda_k(\mathbf{x}_s^n, \mathbf{x}_f^n)$. Then the semi-implicit friction force is integrable with the friction potential computed as

$$D_{\mathrm{sf}}(\mathbf{x}_s, \mathbf{x}_f) = \sum_{k \in \mathcal{A}^n} \mu_t \lambda_k^n f_0(\|\mathbf{u}_k\|), \tag{4.21}$$

where f_0 is defined by the relation $f_0' = f_1$ and $\mathcal{A}^n$ is the set containing all activate particle-triangle contact pairs at the previous time step n.

4.5 Optimization Time Integrator

With the above potential energies modeling all the solid and fluid forces, now we can build a unified two-way solid-fluid coupling framework. By stacking all the nodal positions and velocities of the SPH particles and FEM nodes as $\mathbf{x} = [\mathbf{x}_f^T, \mathbf{x}_s^T]^T$ and $\mathbf{v} = [\mathbf{v}_f^T, \mathbf{v}_s^T]^T$, we define $\Psi(\mathbf{x}) = \Psi_s(\mathbf{x}_s)$, $P(\mathbf{x}) = P_I(\mathbf{x}_f) + P_V(\mathbf{x}_f)$, and $C_{\mathrm{sf}}(\mathbf{x}) = B_{\mathrm{sf}}(\mathbf{x}_s, \mathbf{x}_f) + D_{\mathrm{sf}}(\mathbf{x}_s, \mathbf{x}_f)$. Combined with the solid-solid contact potential $C_{\mathrm{ss}}(\mathbf{x})$ from IPC, our solid-fluid coupling problem can be solved in a monolithic manner by applying implicit Euler time integration

$$\begin{cases} \mathbf{v}^{n+1} = \mathbf{v}^n + h\mathbf{M}^{-1}(\mathbf{f}_{\text{ext}} - \nabla P(\mathbf{x}^{n+1}) - \nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1})), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases} \tag{4.22}$$

which is equivalent to

$$\mathbf{x}^{n+1} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \hat{\mathbf{x}}^n\|^2_{\mathbf{M}} + h^2(P(\mathbf{x}) + \Psi(\mathbf{x}) + C(\mathbf{x})) \tag{4.23}$$

with the mass matrix $\mathbf{M}$, time step size h, predictive position $\hat{\mathbf{x}}^n = \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$, and total contact potential $C(\mathbf{x}) = C_{\text{sf}}(\mathbf{x}) + C_{\text{ss}}(\mathbf{x})$.

CHAPTER 5

Efficient Solver

A straightforward way to robustly solve the time-stepping optimization problem (4.23) is to apply the projected Newton's method with line search (Li et al., 2020). At every iteration, the search direction $\mathbf{p}$ can be computed by solving the linear system

$$\begin{bmatrix} \mathbf{H}_f & \mathbf{G} \\ \mathbf{G}^T & \mathbf{H}_s \end{bmatrix} \mathbf{p} = \begin{bmatrix} \mathbf{g}_f \\ \mathbf{g}_s \end{bmatrix}, \tag{5.1}$$

where $\mathbf{H}_f$ and $\mathbf{H}_s$ are the (projected) Hessian matrices with respect to the position of fluids and solids, respectively, and $\mathbf{G} = \frac{\partial^2 E}{\partial \mathbf{x}_f \partial \mathbf{x}_s}$ denotes the coupling submatrix. Unfortunately, solving this linear system can be a severe bottleneck in practice. One reason is that SPH techniques need sufficiently many neighbors to accurately approximate physical quantities, which results in a much larger and denser fluid Hessian matrix $\mathbf{H}_f$ compared to the solid one. In addition, the optimization may require many iterations to converge due to the sharpness of barrier energy, especially in contact-rich cases.

Since our fluid energies are all quadratic, we separate them from the highly nonlinear solids and contact energies via a robust time splitting scheme (Section 5.1) so that the fluid part can be solved within a single Newton iteration per time step. We then propose efficient methods to solve the domain-decomposed linear systems (Section 5.2).

5.1 Time Splitting

5.1.1 Baseline Time Splitting

Intuitively, we can split the original time integration into a **fluid phase**

$$\begin{cases} \mathbf{v}_f^{n+1/2} = \mathbf{v}_f^n + h\mathbf{M}_f^{-1}\left(-\nabla_f P\left([(\mathbf{x}_f^{n+1/2})^T, (\mathbf{x}_s^n)^T]^T\right) + \mathbf{f}_f\right), \\ \mathbf{x}_f^{n+1/2} = \mathbf{x}_f^n + h\mathbf{v}_f^{n+1/2}, \end{cases} \tag{5.2}$$

and a solid-coupling phase

$$\begin{cases} \mathbf{v}^{n+1} = \begin{bmatrix} \mathbf{v}_f^{n+1/2} \\ \mathbf{v}_s^n \end{bmatrix} + h\mathbf{M}^{-1}\left(-\nabla\Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \begin{bmatrix} \mathbf{0} \\ \mathbf{f}_s \end{bmatrix}\right), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases} \tag{5.3}$$

where $\mathbf{f}_f$ and $\mathbf{f}_s$ are the external forces on the fluids and the solids, respectively. In the fluid phase, we solve for an intermediate state for the fluid particles in a single Newton iteration, ignoring contact. Then the highly nonlinear barrier force is resolved in the solid-coupling phase along with elasticity, where the fluid Hessian $\mathbf{H}_f$ reduces to a block-diagonal matrix $\frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_f^2}$. In this setting, nonlinear optimization only happens for fluid boundaries and solid DOFs in the solid-coupling phase. The details of this *Baseline Time Splitting Scheme* can be found in Appendix B.1.

Although this baseline splitting strategy indeed brings a significant performance gain, severe instabilities can occur at the solid-fluid interface if the time step size is not sufficiently small, especially when simulating viscous fluids (Figure 6.3). For example, fluid particles may stick to the solid boundaries. This artifact is also evident in existing SPH fluid solvers, and it is typically addressed by sampling particles at solid boundaries to exert boundary pressures (Becker et al., 2009b; Ihmsen et al., 2010; Akinci et al., 2012). In light of this, we consistently augment the fluid phase with proxy forces for solid-fluid contact to improve stability while avoiding any particle-sampling overhead.

5.1.2 Time Splitting with Contact Proxy

We introduce a solid-fluid contact proxy energy $\hat{C}_{\text{sf}}(\mathbf{x})$ into the fluid phase to efficiently exert approximated interaction forces between the boundaries of solids and fluids. In the ensuing discussion, we will also write contact energy $C(\mathbf{x})$ as the sum of the solid-fluid part $C_{\text{sf}}(\mathbf{x})$ and the solid-solid part $C_{\text{ss}}(\mathbf{x})$ for clarity. To ensure consistency with the original PDE, we cancel the contribution of this contact proxy in the solid-coupling phase. The resulting time integration becomes

$$\begin{aligned}&\begin{cases}\mathbf{v}^{n+1/2} = \mathbf{v}^n + h\mathbf{M}^{-1}(-\nabla P(\mathbf{x}^{n+1/2}) + \mathbf{f}_{\text{ext}} - \nabla \hat{C}_{\text{sf}}(\mathbf{x}^{n+1/2})),\\ \mathbf{x}^{n+1/2} = \mathbf{x}^n + h\mathbf{v}^{n+1/2};\end{cases}\quad ,\\ &\begin{cases}\mathbf{v}^{n+1} = \mathbf{v}^{n+1/2} + h\mathbf{M}^{-1}(-\nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \nabla \hat{C}_{\text{sf}}(\mathbf{x}^{n+1})),\\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1},\end{cases}\end{aligned} \tag{5.4}$$

where the fluid phase now also implicitly updates the solid boundary nodes near the fluids to an intermediate state.

For $\hat{C}_{\text{sf}}(\mathbf{x})$, a straightforward choice is $\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2}C_{\text{sf}}(\mathbf{x})$. But to ensure our fluid phase only contains linear forces, we apply the second-order Taylor expansion of $\frac{1}{2}C_{\text{sf}}(\mathbf{x})$ at $\mathbf{x}^n$ for the approximation in the fluid phase; i.e.,

$$\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2}\left(C_{\text{sf}}(\mathbf{x}^n) + \nabla C_{\text{sf}}(\mathbf{x}^n)(\mathbf{x} - \mathbf{x}^n) + \frac{1}{2}\|\mathbf{x} - \mathbf{x}^n\|^2_{\nabla^2 C_{\text{sf}}(\mathbf{x}^n)}\right), \tag{5.5}$$

while in the solid-coupling phase, we simply use $\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2}C_{\text{sf}}(\mathbf{x})$. In Appendix B.2, we prove that our time splitting scheme with contact proxy only has an $\mathcal{O}(h^4)$ mismatch compared to the implicit Euler solution. Reformulating both phases (5.4) as optimization problems, we obtain

$$\begin{aligned}\mathbf{x}^{n+1/2} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \hat{\mathbf{x}}^n\|^2_{\mathbf{M}} + h^2(P(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x})),\\ \mathbf{x}^{n+1} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{x}^{n+1/2}\|^2_{\mathbf{M}} + h^2(\Psi(\mathbf{x}) + \frac{1}{2}C_{\text{sf}}(\mathbf{x}) + C_{\text{ss}}(\mathbf{x})),\end{aligned} \tag{5.6}$$

Algorithm 1: Time Splitting with Contact Proxy

```
$\mathbf{x} \leftarrow \mathbf{x}^n, \ \hat{\mathbf{x}}^n \leftarrow \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$
SPH Neighbor Search & Density Update
$\hat{C}_{\text{sf}}(\mathbf{x}) \leftarrow$ 2nd Taylor Expansion of $\frac{1}{2}C_{\text{sf}}(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}^n$
// Fluid Phase
$\mathbf{H} \leftarrow h^2\left(\nabla^2 P(\mathbf{x}) + \nabla^2\hat{C}_{\text{sf}}(\mathbf{x})\right) + \mathbf{M}$
$\mathbf{p} \leftarrow -\mathbf{H}^{-1}\left(h^2(\nabla P(\mathbf{x}) + \nabla\hat{C}_{\text{sf}}(\mathbf{x})) + \mathbf{M}(\mathbf{x} - \hat{\mathbf{x}}^n)\right)$
$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{p}$
$\mathbf{x}^{n+1/2} \leftarrow \mathbf{x}$
// Solid-Coupling Phase
do
    $\mathbf{H} \leftarrow h^2\left(\nabla^2\Psi(\mathbf{x}) + \frac{1}{2}\nabla^2 C_{\text{sf}}(\mathbf{x}) + \nabla^2 C_{\text{ss}}(\mathbf{x})\right) + \mathbf{M}$
    $\mathbf{g} \leftarrow h^2\left(\nabla\Psi(\mathbf{x}) + \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}) + \nabla C_{\text{ss}}(\mathbf{x})\right) + \mathbf{M}(\mathbf{x} - \mathbf{x}^{n+1/2})$
    $\mathbf{p} \leftarrow -\mathbf{H}^{-1}\mathbf{g}$
    $\alpha \leftarrow$ Backtracking Line Search with CCD
    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$
while $\frac{1}{h}\|\mathbf{p}\| > \epsilon$
$\mathbf{x}^{n+1} \leftarrow \mathbf{x}, \ \mathbf{v}^{n+1} \leftarrow (\mathbf{x} - \mathbf{x}^n)/h$
return $\mathbf{x}^{n+1}, \mathbf{v}^{n+1}$
```

where $\hat{\mathbf{x}}^n = \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$ with $\mathbf{f}_{\text{ext}} = [\mathbf{f}_f^T, \mathbf{f}_s^T]^T$.

In addition to avoiding fluid particle sticking issues without extra computationally expensive costs, another benefit of our method is that it helps reduce the number of Newton iterations needed to solve the problem. Typically, the barrier method takes many Newton iterations to resolve high-speed impacts. With our scheme, when high-speed fluid particles are colliding with a deformable object, their speed will be significantly reduced after the fluid phase due to the contact proxy. The reduced speed will then be taken into the solid-coupling phase, which makes the nonlinear optimization easier to solve (by having fewer contact constraint set changes). The details of our proxy-based time splitting scheme are found in Algorithm 1.

Similarly, one can also separate elasticity from contact energy using the contact proxy. In this fashion, we would have a three-phase (fluid, solid, and contact) time splitting scheme

$$
\begin{aligned}
\mathbf{x}^{n+1/3} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \hat{\mathbf{x}}^n\|_{\mathbf{M}}^2 + h^2(P(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x})), \\
\mathbf{x}^{n+2/3} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{x}^{n+1/3}\|_{\mathbf{M}}^2 + h^2(\Psi(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x}) + \hat{C}_{\text{ss}}(\mathbf{x})), \\
\mathbf{x}^{n+1} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{x}^{n+2/3}\|_{\mathbf{M}}^2 + h^2(\frac{1}{3}C_{\text{sf}}(\mathbf{x}) + \frac{1}{2}C_{\text{ss}}(\mathbf{x})),
\end{aligned}
\tag{5.7}
$$

where $\hat{C}_{\text{sf}}(\mathbf{x})$ and $\hat{C}_{\text{ss}}(\mathbf{x})$ are the second-order Taylor expansions of $\frac{1}{3}C_{\text{sf}}(\mathbf{x})$ and $\frac{1}{2}C_{\text{ss}}(\mathbf{x})$, respectively. However, this aggressive splitting scheme only applies to inversion-robust constitutive models; e.g., the fixed corotated model (Stomakhin et al., 2012). While inversion may be guaranteed to be prevented at the solid phase where the elasticity energy is considered, this may not hold in the contact phase. Despite this limitation, the three-phase splitting scheme can still work properly for inversion-robust constitutive models in practice to further accelerate the simulation.

5.2 Solving Linear Systems

In our time splitting scheme, solving large sparse linear systems dominates both the computational and memory costs of each phase. We thus devise matrix-free and Schur-complement-based strategies to solve them efficiently.

5.2.1 Fluid Phase

Since 2-ring neighbors of SPH particles need to be considered in our formulation, both constructing and directly factorizing the Hessian matrix can cost a significant amount of time and memory. Therefore, we devise a matrix-free conjugate gradient (CG) solver to efficiently solve for the intermediate state of fluids.

As all energy potentials are quadratic in this phase, the energy gradient $\mathbf{g}(\mathbf{x})$ is merely a linear function of $\mathbf{x}$ with constant coefficient matrix $\mathbf{H}(\mathbf{x})$. Thus, the product between $\mathbf{H}(\mathbf{x})$ and an arbitrary vector $\mathbf{p}$ can be expressed as

$$
\mathbf{H}(\mathbf{x})\mathbf{p} = \mathbf{g}(\mathbf{p}) - \mathbf{g}(\mathbf{0}). \tag{5.8}
$$

This allows us to compute gradients to evaluate the matrix-vector product, and we only need to acquire the 3×3 diagonal blocks of the Hessian for block-Jacobi preconditioning in our CG solver.

5.2.2 Solid-Coupling Phase

As the fluid energy potential is not included in this phase, the components of the Hessian matrix become

$$\mathbf{H}_f = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_f^2}, \quad \mathbf{G} = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_s \partial \mathbf{x}_f}, \quad \mathbf{H}_s = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_s^2} + \frac{\partial^2 \Psi(\mathbf{x})}{\partial \mathbf{x}_s^2}. \tag{5.9}$$

Although this linear system is no longer that intractable, it is not optimal to directly factorize the whole system given the considerable amount of nonzeros in $\mathbf{H}_f$ and $\mathbf{G}$ when fluid resolution is high.

Thus, we design a domain decomposed linear solver that treats $\mathbf{H}_f$ and $\mathbf{H}_s$ separately. Based on the Schur complement (Zhang, 2006), the inverse of our Hessian matrix can be expressed as

$$\mathbf{H}^{-1} = \begin{bmatrix} \mathbf{H}_f^{-1} + \mathbf{H}_f^{-1}\mathbf{G}(\mathbf{H}/\mathbf{H}_f)^{-1}\mathbf{G}^T\mathbf{H}_f & -\mathbf{H}_f^{-1}\mathbf{G}(\mathbf{H}/\mathbf{H}_f)^{-1} \\ -(\mathbf{H}/\mathbf{H}_f)^{-1}\mathbf{G}^T\mathbf{H}_f & (\mathbf{H}/\mathbf{H}_f)^{-1} \end{bmatrix}, \tag{5.10}$$

where $\mathbf{H}/\mathbf{H}_f = \mathbf{H}_s - \mathbf{G}^T\mathbf{H}_f^{-1}\mathbf{G}$ is the Schur complement of block $\mathbf{H}_f$. Since the nonzeros of $\mathbf{H}_f$ exist only in the diagonal blocks, it is trivial to obtain the inverse matrix $\mathbf{H}_f^{-1}$. We can then apply the CHOLMOD (Chen et al., 2008) LLT solver to factorize $\mathbf{H}/\mathbf{H}_f$, the dimensionality of which is on the order of the number of solid DOFs, and then the search direction can be computed via matrix-vector products and back-solves. When there is no solid-fluid interaction, the sparsity pattern of $\mathbf{H}/\mathbf{H}_f$ remains identical to that of $\mathbf{H}_s$. Only when two solid nodes i and j are interacting with the same fluid particle will, in 3D, the 3×3 block $(\mathbf{H}/\mathbf{H}_f)_{i,j}$ become non-zero. Typically, this happens only for neighboring mesh primitives, and thus the sparsity pattern of $\mathbf{H}/\mathbf{H}_f$ is mostly nice.

Note that when the three-phase time splitting scheme (5.7) is used, our domain decomposed solver can also be applied to the solid and contact phases, since their systems share a similar structure with the solid-coupling phase.

CHAPTER 6

Experiments

Our code is implemented in C++ with Eigen for basic linear algebra operations and Intel's TBB for multi-threading. The time step size of all our simulations is adaptively chosen by the SPH CFL condition and a user-defined upper bound. We set the support radius of our SPH kernel function to $2d$, where d is the particle diameter. In our implementation, we use the cubic Spline kernel for density estimation and the Spiky kernel for gradient calculation. For the results shown in Figures 6.10, 6.8, 6.5, and 6.2, we employ our three-phase time splitting scheme, demonstrating its efficacy when the constitutive models are compatible with mesh inversion. For the remaining simulations, we stick with our two-phase time splitting scheme. Most experiments were performed on a 24-core 3.50GHz Intel i9-10920X workstation, except for the comparative study with ElastoMonolith (Takahashi and Batty, 2022). We demonstrate that our method achieves efficient and robust solid-fluid coupling. The parameters and timing breakdown of our simulations are specified in Table 6.1 and Figure 6.1, respectively.

6.1 Ablation Study

6.1.1 Time Splitting Evaluation

Three simulations (Figure 6.2 and Figure 6.3) are performed to demonstrate the efficiency of time splitting and the efficacy of our contact proxy in maintaining stability.

To begin, we must choose a proper time step h. First, it must be restricted by the CFL condition, otherwise severe volume loss may be observed due to the SPH approximation error. Additionally, in contrast to the joint optimization (4.23), the time splitting scheme

Scene	N_{fluid}	N_{solid}	k_I	ν_f	d	ρ_f	E	ν_s	ρ_s	T
Fig. 6.2	97K	2.3K	2×10^5	0	15	1000	1×10^5	0.3	500	0.3
Fig. 6.3a	238K	0	1×10^5	100	10	1200	-	-	-	0.4
Fig. 6.5	103K	16K	1×10^5	0	10	1000	1×10^5	0.3	200	1.3
Fig. 6.6	280K	0	2×10^5	0.005	25	1000	-	-	-	0.4
Fig. 6.7a	52K	3.7K	1×10^5	0	10	1000	4×10^3	0.49	200	0.4
Fig. 6.7b	101K	4.5K	6×10^4	0	6.4	1000	1×10^3	0.49	200	1.0
Fig. 6.8	787K	66K	2×10^5	1	10	1000	1×10^5	0.4	200/700/1200	5.9
Fig. 6.10*	486K	12K	4×10^4	0	5	1000	-	-	500	7.9
Fig. 6.11	159K	9K	3×10^4	25	3	1000	5×10^8	0.49	1000	1.8
Fig. 6.12*	789K	13K	1×10^5	0.2	10	1000	1×10^5	0.45	100/700	4.9
Fig. 6.13*	1M	43K	2.5×10^5	0.1	25	1000	-	-	500	37.9

Table 6.1: Simulation statistics including the number of fluid particles N_{fluid}, number of solid vertices N_{solid}, incompressibility coefficient k_I, $[Pa]$, dynamic viscosity ν_f, $[Pa{\cdot}s]$, fluid particle diameter d, $[mm]$, fluid density ρ_f, $[kg/m^3]$, Young's modulus E, $[Pa]$, Possion's ratio ν_s, solid density ρ_s, $[kg/m^3]$, and the average simulation time T, $[min]$ for each frame. Timing statistics are measured on a 24-core 3.50GHz Intel i9-10920X machine, except for Figure 6.7, which is tested on the "e2-standard-8" (8 cores with 32GB RAM) Google Compute Engine. Note that examples marked * contain codimensional materials whose parameter settings are not covered here.

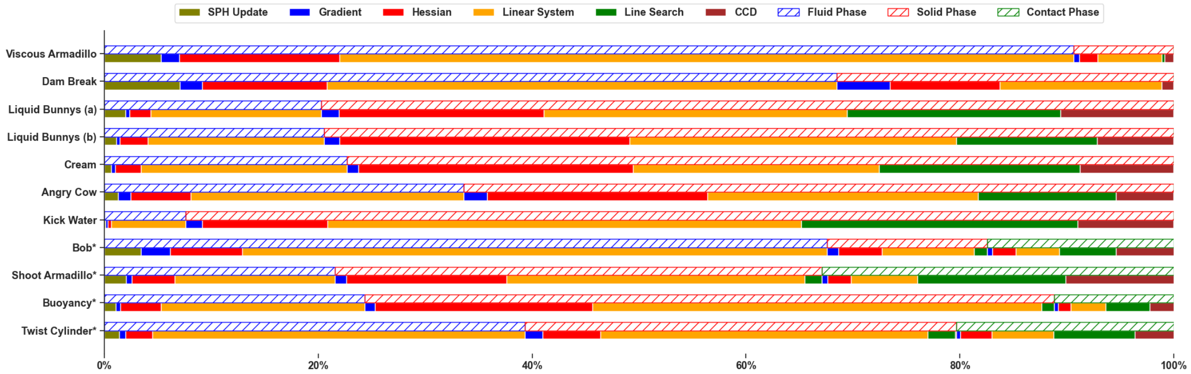


Figure 6.1: Timing breakdown. We show the timing profile of different simulation phases and plot the proportions of the major routines. Examples marked with * are simulated using our three-phase time splitting scheme. Other examples are generated with the two-phase scheme. In particular, SPH update (including neighborhood search and density update) only occurs in the fluid phase, line search happens in the solid and contact phases for non-linear optimization, and continuous collision detection (CCD) is counted when IPC contact energy is considered.

Scene	Scheme	Sec/Frame	# Newton Iter./Frame
Fig. 6.2	Joint/TS/TSCP	66.1 / 38.0 / 22.5	63.5 / 117.3 / 37.1
Fig. 6.3a	Joint/TS/TSCP	41.3 / 32.3 / 25.5	16.5 / 29.0 / 10.5

Table 6.2: Statistics of different time stepping schemes. Joint Optimization (Joint), Baseline Time Splitting (TS), and Time Splitting with Contact Proxy (TSCP). Our TSCP is much faster than both the Joint and TS schemes.

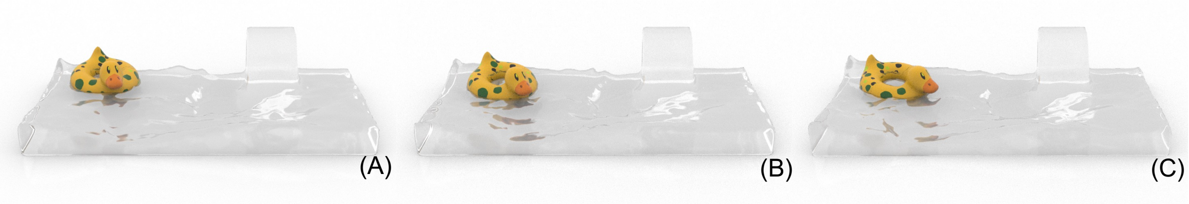
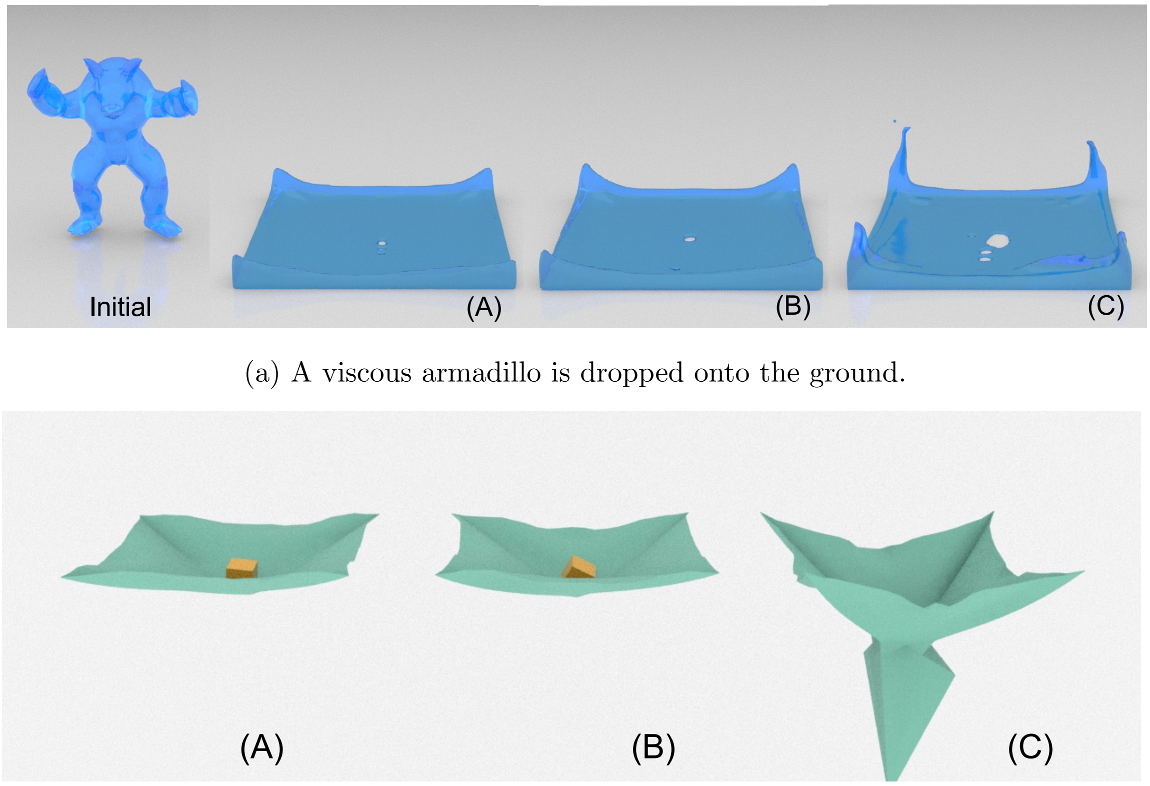


Figure 6.2: Bob simulated with (A) Joint Optimization, (B) Time Splitting with Contact Proxy, and (C) Baseline Time Splitting. For this example, baseline time splitting can also produce visually plausible results, and our proxy-assisted scheme is 3× faster than joint optimization.

usually requires smaller time steps to remain stable, which imposes a second time step constraint. However, we observed that, in practice, even using the largest CFL time step, our proxy-assisted time splitting can still work properly and produce stable simulation results. Hence, for comparison, we use the largest CFL time step for both schemes to maximize their performance, as smaller h typically takes more Newton iterations in total to simulate a frame. For the joint optimization, since direct factorization is intractable, we solve (5.1) using the block-Jacobi preconditioned conjugate gradient solver with the fluid part of the matrix free.

As shown in Table 6.2, even in these simple examples, our time splitting scheme is significantly, up to 3× faster than joint optimization, especially for cases involving contacts between fluids and deformable solids (e.g., Figure 6.2). This improvement stems from no longer having to solve for the incompressibility of fluids repeatedly within a time step. Moreover, one can also find that much fewer Newton iterations are needed with our proxy-assisted time splitting scheme. As discussed in Section 5.1.2, this is because the challenging high-speed impacts are already partially resolved in the fluid phase. Another benefit of time splitting is the support of different error tolerances for the two phases.



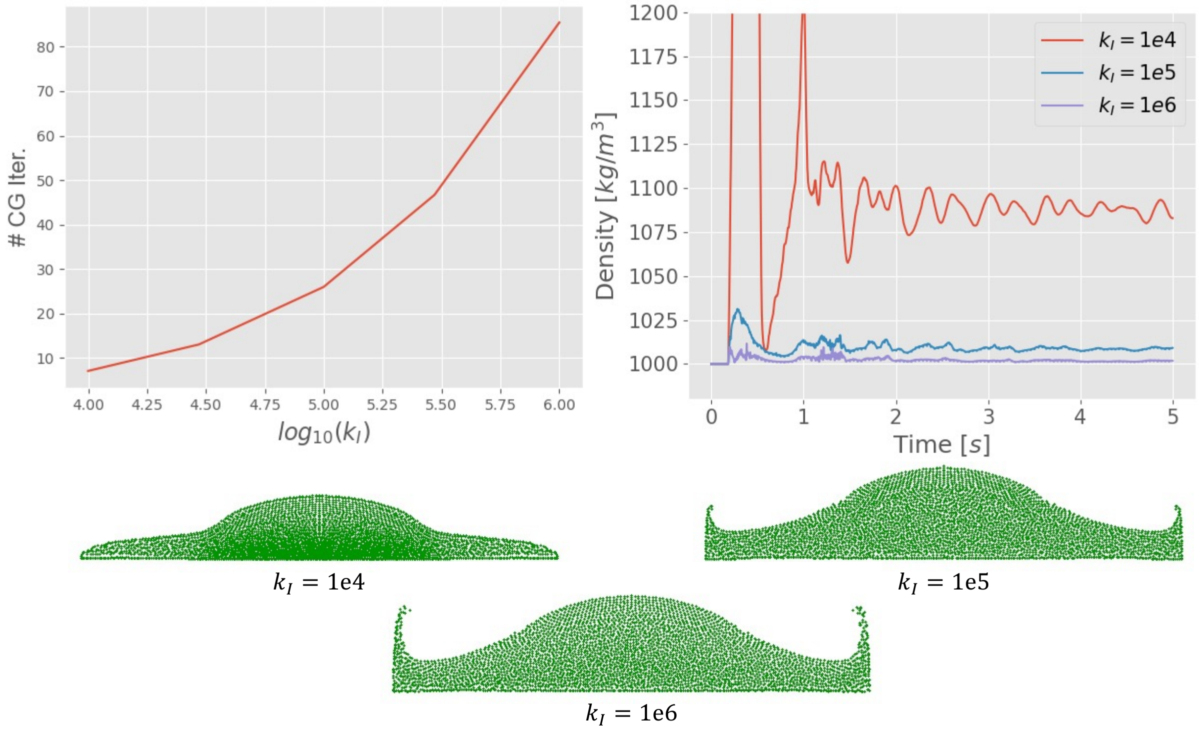
(a) A viscous armadillo is dropped onto the ground.

(b) Cube on cloth. An elastic cube is dropped onto a square cloth with four corners fixed.

Figure 6.3: Simulation results of (A) Joint Optimization, (B) Time Splitting with Contact Proxy, and (C) Baseline Time Splitting. While directly applying time splitting results in instability at the boundaries, our results with the contact proxy are consistent with joint optimization.

Errors in the fluid phase are sourced from the solution deviation of the CG solver, while in the solid phase they are directly controlled by the tolerance of Newton's method. Typically, setting a slightly higher tolerance for fluids yields better performance while still producing visually convincing results.

Aside from its efficiency, our proposed contact proxy also improves the stability of the time splitting scheme. Alhough the simulation results of the baseline time splitting scheme look fine in the case of inviscid fluids, matters get worse when it is applied to viscous fluids. In Figure 6.3a, a viscous armadillo is dropped to the ground. In this example, the baseline time splitting scheme produces severe sticky artifacts at the boundary, and the fluid surface cannot eventually calm down. By consistently applying our contact proxy to exert boundary pressure in the fluid phase, the artifacts can be well resolved, as



k_I = 1e4

k_I = 1e5

k_I = 1e6

Figure 6.4: Statistics of simulations with different values of stiffness parameter k_I. A larger k_I preserves volume better, but necessitates more CG iterations. In this case, a proper $k_I = 10^5$ Pa can be set to balance the computational cost and visual artifacts.

demonstrated in Figure 6.3a. Similarly, our idea of contact proxy is also applicable to further separate elasticity from IPC contact while maintaining stability, leading to our three-phase scheme (Figure 6.3b).

6.1.2 Linear Solver Evaluation

For the fluid phase, we designed a matrix-free conjugate gradient (CG) solver that calculates the matrix-vector product via a gradient computation to avoid the expensive computational and memory costs of direct factorization (Section 5.2.1). However, the performance improvement from this approach is less significant if the number of CG iterations required for convergence is too large, making the cost of computing gradients higher than constructing the Hessian once. In our fluid phase, the number of CG iterations is proportional to the stiffness k_I of the incompressibility energy. A larger k_I can better preserve the volume of the fluids, but also results in a worse-conditioned system, demanding

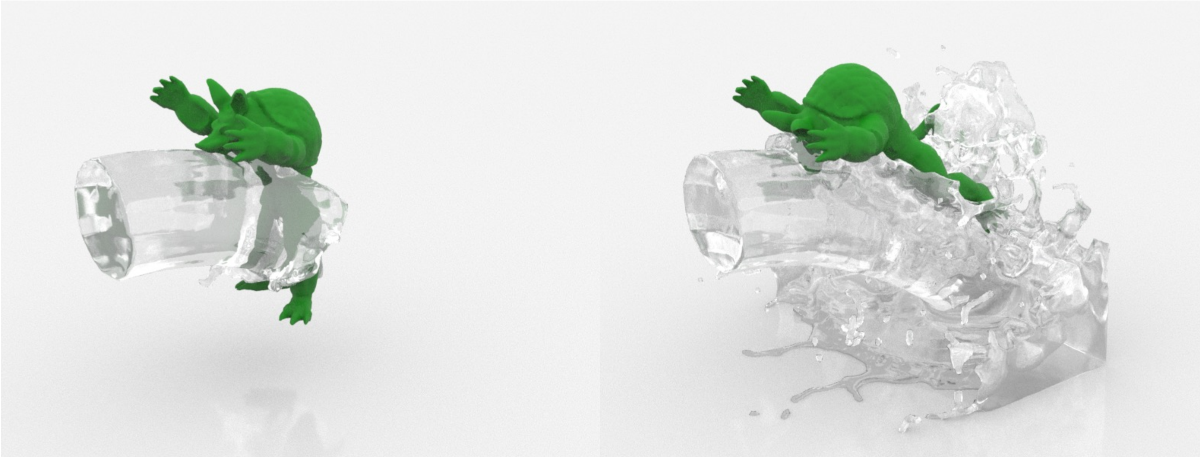


Figure 6.5: Shooting an armadillo with a high-speed water jet.

Solver	Fluid Phase		Solid Phase	Contact Phase	Mem.
	hess	solve	solve	solve	
CG + LLT	14.9	0.49	1.45	0.43	12375
Ours	0.15	0.59	1.11	0.25	1469

Table 6.3: Time and memory cost of different solvers in example 6.5. The costs are measured per time step in units [s] and [MB], respectively. The baseline method uses the Conjugate Gradient (CG) method for the fluid phase and CHOLMOD LLT for the solid and contact phases. Instead, our method employs a matrix-free CG solver for the fluid phase and a domain-decomposed solver for the solid and contact phases, thereby improving efficiency and saving memory.

more iterations to converge (Figure 6.4). In practice, by setting k_I to a proper value, we can efficiently solve the systems within 50 CG iterations without obvious fluid volume losses.

We test the performance of our matrix-free CG solver together with the domain-decomposed solver that we designed for the solid-coupling phase using an armadillo shooting example (Figure 6.5), and present our results in Table 6.3. Our matrix-free CG solver significantly boosts efficiency (20× faster) and reduces memory costs by avoiding the construction of the Hessian matrix. On the other hand, our domain decomposed solver is 40% faster than directly factorizing the solid and contact systems.

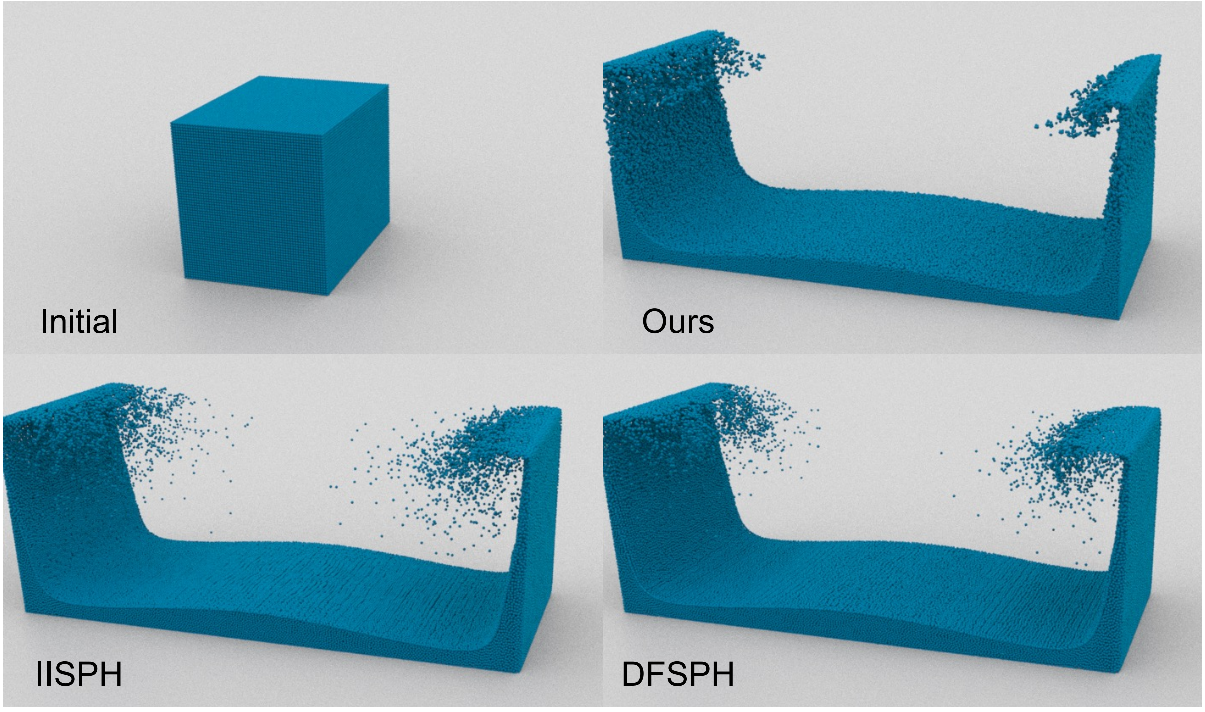


Figure 6.6: Dam break with 280K SPH particles. Our weakly compressible formulation produces stable fluid dynamics without visually evident volume loss. Compared to incompressible SPH solvers IISPH (Ihmsen et al., 2013) and DFSPH (Bender and Koschier, 2015), our simulation results demonstrate more smooth particle distribution.

6.2 Comparisons

In this section, we compare our method with several popular SPH fluid solvers and a state-of-the-art solid-fluid coupling method, ElastoMonolith (Takahashi and Batty, 2022). We leveraged the open-source library SPlisHSPlasH[1] to implement the SPH fluid simulators. To compare our method with ElastoMonolith, we set up two scenes from the aforecited publication with identical parameters and ran all the simulations using an "e2-standard-8" (8 cores with 32GB RAM) Google Compute Engine for fairness.

6.2.1 Fluid Dynamics

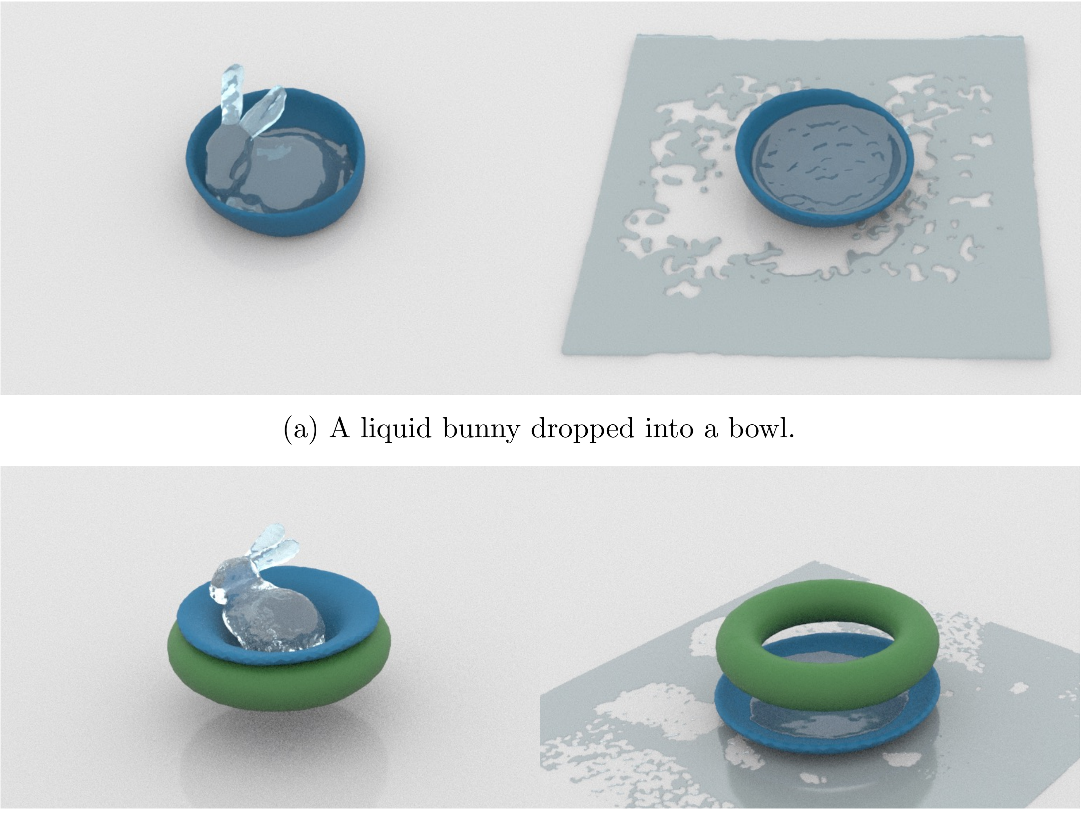
While most existing SPH fluid solvers focus on incompressible fluids, our formulation treats fluids as weakly compressible, allowing us to couple fluids with deformable solids

[1] https://github.com/InteractiveComputerGraphics/SPlisHSPlasH

in a unified framework. We ran a dam break simulation to compare our method with two SPH fluid solvers, IISPH (Ihmsen et al., 2013) and DFSPH (Bender and Koschier, 2015). These methods typically use particle resampling (Akinci et al., 2012, 2013) or implicit representation (Koschier and Bender, 2017; Bender et al., 2019) to exert boundary counter-forces. Our method instead employs IPC (Li et al., 2020) for more robust solid-fluid coupling, with a penetration-free guarantee. We uniformly enforce the same CFL condition for all methods along with an upperbound at $5\,ms$, and use the volume map (Bender et al., 2019) for their boundary handling. As shown in Figure 6.6, although our formulation does not strictly enforce incompressibility, it produces natural fluid dynamics without visually observable volume loss. On the other hand, our method (0.45 min/frame) is slower than IISPH (0.31 min/frame) and DFSPH (0.15 min/frame) due to the more sophisticated boundary handling strategy. However, our approach can couple SPH fluids and elastic solids with arbitrary constitutive models, while most existing SPH methods (Peer et al., 2018; Kugelstadt et al., 2021) treat elastic solids as incompressible, which is not generally applicable.

6.2.2 Solid-Fluid Coupling

We now compare our method with ElastoMonolith (Takahashi and Batty, 2022), which couples Eulerian fluids with Lagrangian solids in a monolithic manner. Following their experiment setting, we ran two solid-fluid coupling simulations with identical parameters using our method (Figure 6.7). The timings of our method for these two scenes are 24.1 sec/frame and 62.8 sec/frame, respectively, both of which are over 5× faster than ElastoMonolith according to their reported timings (253.2 sec/frame and 352.0 sec/frame). Coupling Eulerian fluids with Lagrangian solids requires dealing with geometric differences and SPD reformulation is often needed to make the linear system tractable. As stated in the publication on ElastoMonolith, this SPD reformulation can introduce many additional non-zeros into the system, especially when contacts are rich and solids are intricately shaped. By contrast, our method treats solids and fluids from a unified Lagrangian viewpoint, where solid-solid and solid-fluid contacts are resolved in a unified manner.



(a) A liquid bunny dropped into a bowl.

(b) A liquid bunny and an elastic bowl dropped onto a static torus.

Figure 6.7: Liquid Bunnies. Compared to ElastoMonolith (Takahashi and Batty, 2022), our method achives an over 5× speedup for both of these examples with exactly the same scene setups.

6.3 Complex Scenarios

We next evaluate the efficiency and robustness of our method in more complicated scenarios.

Buoyancy: We drop three elastic elephants with varying densities into water (1000 kg/m^3) (Figure 6.8). The light grey elephant (200 kg/m^3) floats on the surface, the blue elephant (700 kg/m^3) is about half immersed in the water, and the red elephant (1200 kg/m^3) sinks to the bottom. This demonstrates that our method correctly captures buoyancy behavior.

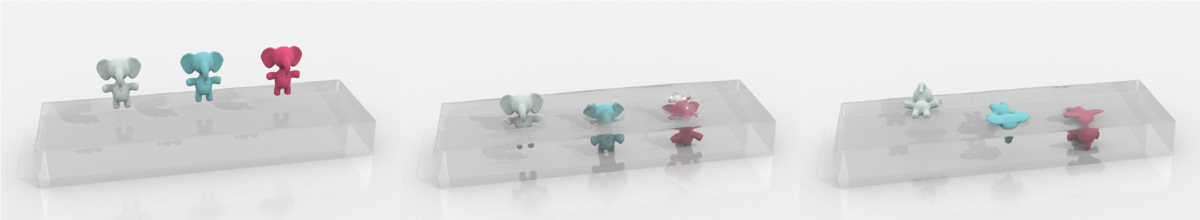


Figure 6.8: Buoyancy. Three elastic elephants with different densities (left to right: 200, 700, and 1200 kg/m^3) fall into the water, demonstrating buoyancy.

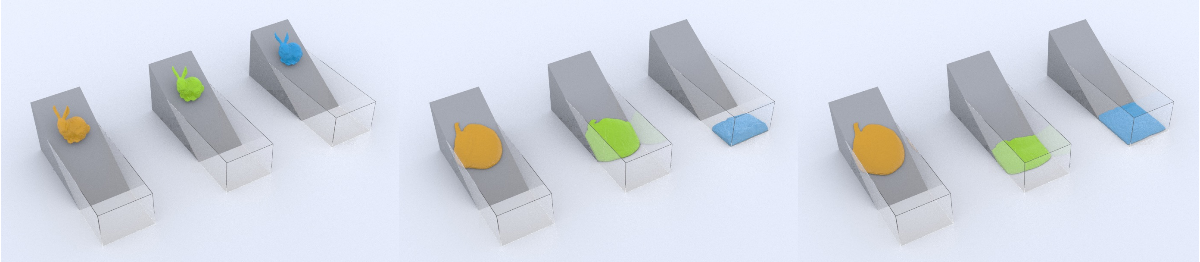


Figure 6.9: Varying friction. Three viscous bunnies are dropped onto the ramp with different coefficients of friction μ (left to right: 0.5, 0.03, 0.0). Our method supports adjustable solid-fluid boundary friction.

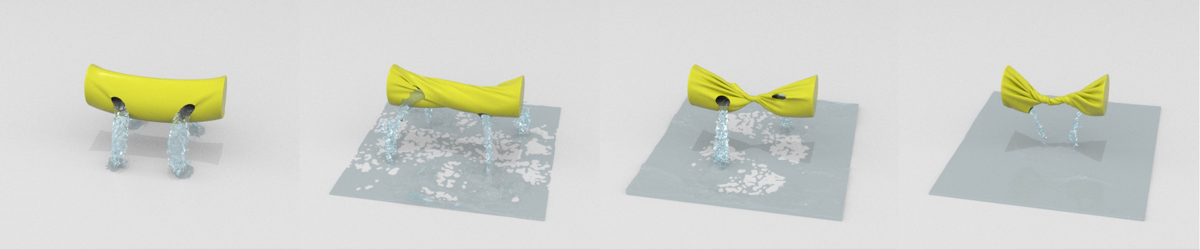


Figure 6.10: Twist cylinder. A cylindrical cloth with four holes is twisted, squeezing out water from the interior.

Varying Friction: We drop three viscous bunnies onto the slope with different coefficients of friction (orange bunny: 0.5, green bunny: 0.03, blue bunny: 0) (Figure 6.9). All three bunnies share the same dynamic viscosity coefficients 100 kg/m^3 and the angle of slope is 30°.

Twist Cylinder: Coupling fluids with thin shells is challenging since, without careful treatment, penetration can easily occur. As stated in (Zarifi and Batty, 2017), Eulerian fluids may flow through solids if their thickness is less than the grid cell size. Conversely, our approach adopts a unified Lagrangian view and IPC guarantees penetration-free

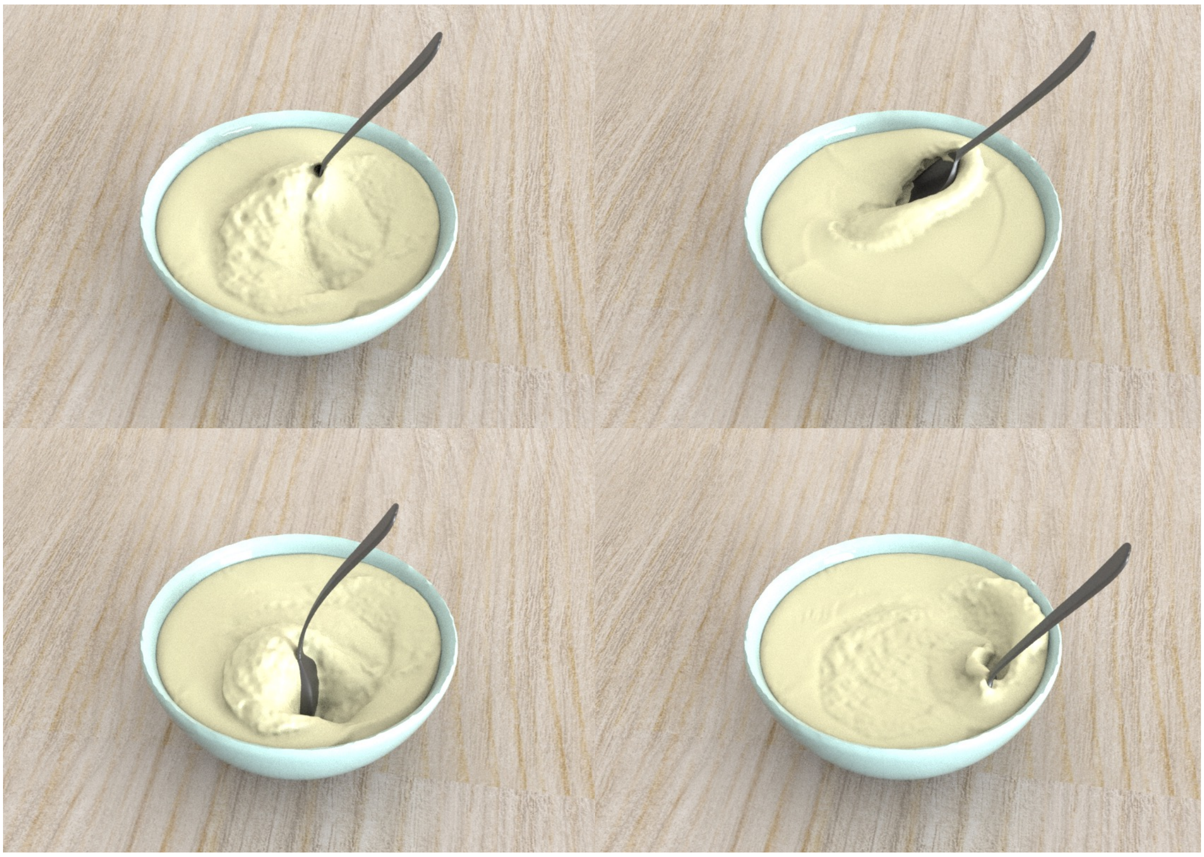


Figure 6.11: Cream is stirred, causing the deformation of the spoon.

results. In the example shown in Figure 6.10, we simulate the twisting of a cylinder full of water. The cylinder is modeled as a thin shell with a $2\,mm$ thickness, and there are two holes in the front and back sides of this cylinder, respectively. The left side and right side are rotated at $72°/s$ and are slowly moved towards each other at $2\,cm/s$. As we twist the cylinder, the water gets squeezed out through the holes. This simulation demonstrates our method produces stable simulation results with a penetration-free guarantee.

Cream: The example of Figure 6.11 exhibits the coupling behaviors of viscous fluids and elastic solids. We use an elastic spoon to stir the cream in a porcelain bowl. The spoon handle rotates around y-axis at $360°/s$ $(0.2\,m/s)$ while the bowl is fixed on the table. As shown in our simulation results, the spoon deforms due to the resistance forces subjected upon it by the viscous cream while stirring.

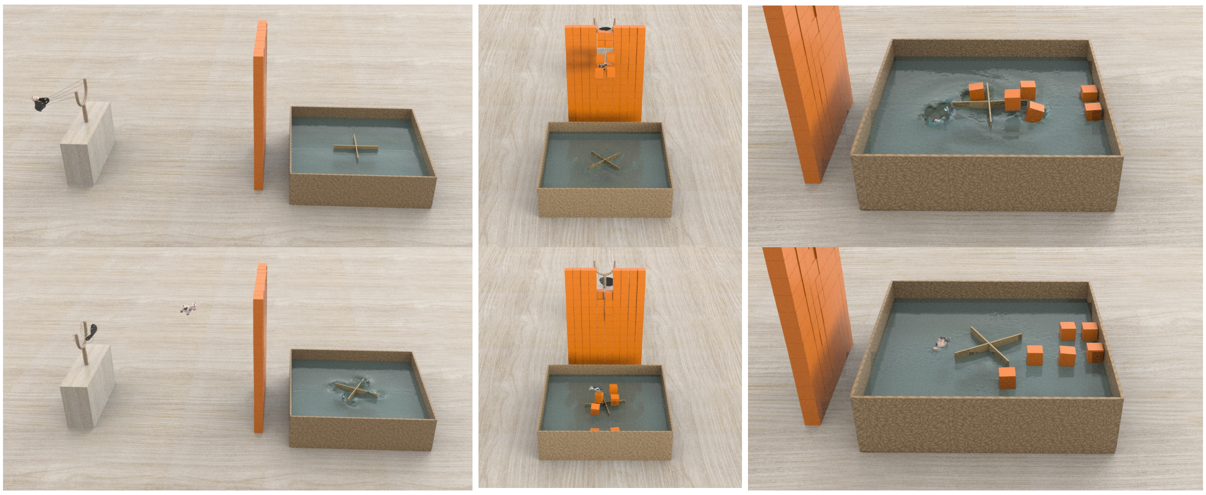


Figure 6.12: Angry cow. We show that our method can simulate the coupling of materials in arbitrary codimensions, including fluid particles, rods (rubber bands), thin shells (leather pad), deformable solids (cow), and rigid bodies (cubes). We launch an angry cow with a slingshot, and the cow penetrates through the wall and then falls into the water. All the interactions between the various materials are captured accurately.

Angry Cow: We next show that our framework can simulate natural physical behaviors of geometries in arbitrary codimensions (0, 1, 2, and 3) as well as their interactions. In the scene scene shown in Figure 6.12, the codimensional-0,1,2 objects respectively refer to fluid particles, rubber bands, and the leather pad. A deformable cow is launched by the slingshot, hitting the wall consisting of rigid cubes, and then falling into the water pool, producing interesting physical behaviors. The density of the rigid cubes and the cow are $100\,kg/m^3$ and $700\,kg/m^3$, respectively.

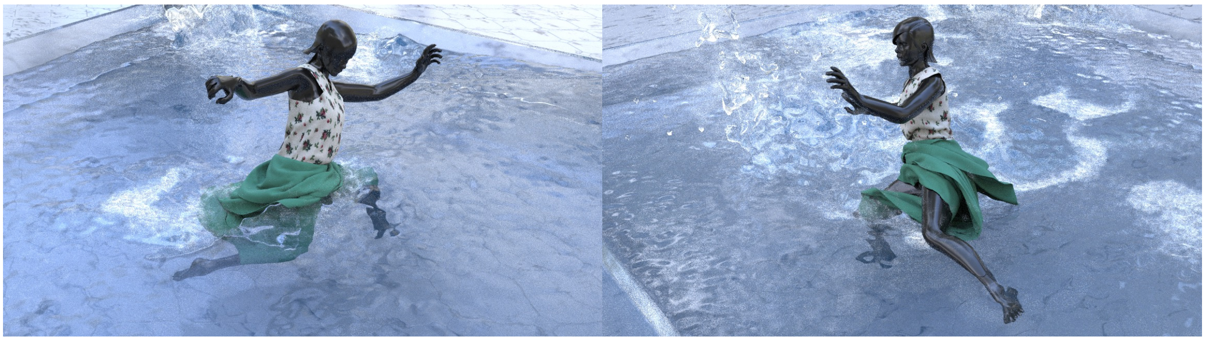


Figure 6.13: Kicking water. Our method accurately captures the complex interactions between the water, the multi-layer skirt, and the mannequin body without any interpenetration as the skirted mannequin kicks in a swimming pool and splashes water.

Kicking Water: In the example of Figure 6.13, we show a scene in which a mannequin dressed in a multilayer skirt kicks in a large pool of water involving complex interactions between fluid particles and garments. As the mannequin kicks vigorously in and above the water, our method simulates the natural deformations of the skirt caused by the contact with the water, and the resulting water splashing is also correctly captured. Our method resolves the contacts among the fluids particles, the thin garments, and the comples, rapidly moving boundaries accurately and with penetration-free guarantees.

CHAPTER 7

Conclusions and Future Work

We have presented a unified two-way strong coupling framework for weakly-compressible SPH fluids and nonlinear elastic FEM solids. To achieve this, we modeled solid-fluid interactions as contact forces between SPH particles and FEM boundary elements, applying the IPC method for guaranteed non-penetration and stability. As we track the volume change of SPH particles in an updated Lagrangian fashion, the incompressibility energy stays quadratic and nice particle distributions are maintained. Utilizing a symmetric approximation of discrete viscosity forces, we proposed a viscosity potential compatible with optimization time integration. We then proposed a time splitting scheme with a contact proxy to efficiently solve the time integration optimization while maintaining robustness. Performance is further boosted by our matrix-free conjugate gradient method and a domain-decomposed solver based on the Schur complement.

Compared to existing works (Zarifi and Batty, 2017; Takahashi and Batty, 2022) coupling Eulerian fluids with Lagrangian elastic solids, our method treats both fluids and solids in a Langrangian manner, avoiding the need to deal with different spatial discretizations. Under such a unified view, our method achieves more convenient and robust two-way coupling, even between fluids and codimensional solids. Likewise, unlike existing SPH methods (Kugelstadt et al., 2021; Peer et al., 2018) that treat all materials as SPH particles, our formulation enjoys both the efficiency of SPH fluids and the accuracy of FEM solids.

There are many worthwhile future research directions. First, when fluid DOFs dominate, building and querying the spatial hash for each fluid particle can manifest a considerable cost. In fact, since there is no solid-fluid contact among interior particles, for better efficiency we can construct the spatial data structure only in the intersection between the extended bounding boxes of the fluids and each solid. Additionally, the adhesion between solids and fluids is also an interesting behavior to model. Similar to the barrier energy, adhesion forces can be exerted on close solid-fluid primitive pairs but in the opposite direction. Modeling adhesion by resolving the surface tension of fluids is also an interesting avenue for future work.

APPENDIX A

Derivatives of Fluid Potentials

The gradient and Hessian of the incompressibility potential with respect to the fluid particle position are

$$\begin{aligned}\frac{\partial P_I(\mathbf{x})}{\partial \mathbf{x}} &= \sum_i k_I V_0 (J_i - 1) J_i^n h \frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}}, \\ \frac{\partial^2 P_I(\mathbf{x})}{\partial \mathbf{x}^2} &= \sum_i k_I V_0 (J_i^n)^2 h^2 \frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}} \left(\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}}\right)^T,\end{aligned} \tag{A.1}$$

where

$$\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}_k} = \begin{cases} \sum\limits_j -\frac{m_j}{h\rho_j^n} \nabla W_{ij} + \sum_b -\frac{m_b}{h\rho_b^n} \nabla_i W_{ib} & \text{for } k = i, \\ \frac{m_j}{h\rho_j^n} \nabla W_{ij} & \text{for } k = j, \\ 0 & \text{otherwise,} \end{cases} \tag{A.2}$$

where $\mathbf{x}_k$ denotes the position of particle k, and $\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}_k}$ can be nonzero if particle k is a neighbor of particle i or $k = i$. According to (A.1), the constant Hessian matrix is obviously positive semi-definite (PSD) since it is simply the sum of the outer product of a vector with positive coefficients.

Similarly, the gradient and Hessian of the viscosity potential with respect to the fluid particle position are

$$\begin{aligned}\frac{\partial P_V(\mathbf{x})}{\partial \mathbf{x}_i} &= \nu \sum_j \mathbf{V}_{ij} \mathbf{v}_{ij}^{n+1}, \\ \frac{\partial^2 P_V(\mathbf{x})}{\partial \mathbf{x}_i \partial \mathbf{x}_k} &= \begin{cases} \frac{\nu}{\hat{h}} \sum_j \mathbf{V}_{ij} & \text{for } k = i, \\ -\frac{\nu}{\hat{h}} \mathbf{V}_{ij} & \text{for } k = j, \\ 0 & \text{otherwise,} \end{cases}\end{aligned} \tag{A.3}$$

where $\mathbf{x}_i$ and $\mathbf{x}_k$ denote the positions of particle i and k, respectively. Since $\mathbf{V}_{ij} = -4(d+2)\frac{m_i m_j}{\rho_i+\rho_j}\frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2+0.01\hbar^2} \propto \mathbf{x}_{ij}^n(\mathbf{x}_{ij}^n)^T$ is a 3×3 constant PSD matrix for any particle pair within a time step, the Hessian of the viscosity potential is a constant PSD matrix as well.

APPENDIX B

Time Splitting

B.1 Baseline Time Splitting

Applying time splitting, we can split the original time integration into a fluid phase

$$\begin{cases} \mathbf{v}_f^{n+1/2} = \mathbf{v}_f^n + h\mathbf{M}_f^{-1}\left(-\nabla_f P\left([(\mathbf{x}_f^{n+1/2})^T, (\mathbf{x}_s^n)^T]^T\right) + \mathbf{f}_f\right), \\ \mathbf{x}_f^{n+1/2} = \mathbf{x}_f^n + h\mathbf{v}_f^{n+1/2}, \end{cases} \tag{B.1}$$

and a solid-coupling phase

$$\begin{cases} \mathbf{v}^{n+1} = \begin{bmatrix} \mathbf{v}_f^{n+1/2} \\ \\ \mathbf{v}_s^n \end{bmatrix} + h\mathbf{M}^{-1}\left(-\nabla\Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \begin{bmatrix} \mathbf{0} \\ \\ \mathbf{f}_s \end{bmatrix}\right), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases} \tag{B.2}$$

where $\mathbf{f}_f$ and $\mathbf{f}_s$ are the external forces on the fluids and the solids, respectively. The two phases of this baseline time splitting scheme have equivalent optimization forms

$$\begin{aligned} \mathbf{x}_f^{n+1/2} &= \arg\min_{\mathbf{x}_f} \frac{1}{2}\|\mathbf{x}_f - \hat{\mathbf{x}}_f^n\|_{\mathbf{M}_f}^2 + h^2 P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]^T), \\ \mathbf{x}^{n+1} &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \hat{\mathbf{x}}^{n+1/2}\|_{\mathbf{M}}^2 + h^2(\Psi(\mathbf{x}) + C(\mathbf{x})), \end{aligned} \tag{B.3}$$

where $\hat{\mathbf{x}}_f^n = \mathbf{x}_f^n + h\mathbf{v}_f^n + h^2\mathbf{M}_f^{-1}\mathbf{f}_f$ and $\hat{\mathbf{x}}^{n+1/2} = \mathbf{x}^n + h[(\mathbf{v}_f^{n+1/2})^T, (\mathbf{v}_s^n)^T]^T + h^2\mathbf{M}^{-1}[\mathbf{0}^T, \mathbf{f}_s^T]^T$ with $\mathbf{v}_f^{n+1/2} = (\mathbf{x}_f^{n+1/2} - \mathbf{x}_f^n)/h$. The details of the optimization algorithm can be found in Algorithm 2.

Algorithm 2: Baseline Time Splitting

```
$\mathbf{x} \leftarrow \mathbf{x}^n, \hat{\mathbf{x}}_f^n \leftarrow \mathbf{x}_f^n + h\mathbf{v}_f^n + h^2\mathbf{M}_f^{-1}\mathbf{f}_f$
SPH Neighbor Search & Density Update
// Fluid Phase
$\mathbf{H}_f \leftarrow h^2\nabla_f^2 P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]) + \mathbf{M}_f$
$\mathbf{p}_f \leftarrow -\mathbf{H}_f^{-1}\left(h^2\nabla_f P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]^T) + \mathbf{M}_f(\mathbf{x}_f - \hat{\mathbf{x}}_f^n)\right)$
$\mathbf{x}_f \leftarrow \mathbf{x}_f + \mathbf{p}_f$
$\mathbf{x}_f^{n+1/2} \leftarrow \mathbf{x}_f, \mathbf{v}_f^{n+1/2} = (\mathbf{x}_f - \mathbf{x}_f^n)/h$
// Solid Coupling Phase
$\hat{\mathbf{x}}^{n+1/2} \leftarrow \mathbf{x}^n + h[(\mathbf{v}_f^{n+1/2})^T, (\mathbf{v}_s^n)^T]^T + h^2\mathbf{M}^{-1}[\mathbf{0}^T, \mathbf{f}_s^T]^T$
do
    $\mathbf{H} \leftarrow h^2\left(\nabla^2\Psi(\mathbf{x}) + \nabla^2 C(\mathbf{x})\right) + \mathbf{M}$
    $\mathbf{p} \leftarrow -\mathbf{H}^{-1}\left(h^2(\nabla\Psi(\mathbf{x}) + \nabla C(\mathbf{x})) + \mathbf{M}(\mathbf{x} - \hat{\mathbf{x}}^{n+1/2})\right)$
    $\alpha \leftarrow$ Backtracking Line Search with CCD
    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$
while $\frac{1}{h}\|\mathbf{p}\| > \epsilon$
$\mathbf{x}^{n+1} \leftarrow \mathbf{x}, \mathbf{v}^{n+1} \leftarrow (\mathbf{x} - \mathbf{x}^n)/h$
return $\mathbf{x}^{n+1}, \mathbf{v}^{n+1}$
```

B.2 Error Analysis

The position update of the implicit Euler step and our proxy-enhanced time splitting scheme can be respectively expressed as

$$\begin{aligned}\mathbf{x}_{\text{IE}}^{n+1} &= \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\Big(-\nabla P(\mathbf{x}_{\text{IE}}^{n+1}) \\ &\qquad - \nabla\Psi(\mathbf{x}_{\text{IE}}^{n+1}) - \nabla C_{\text{sf}}(\mathbf{x}_{\text{IE}}^{n+1}) - \nabla C_{\text{ss}}(\mathbf{x}_{\text{IE}}^{n+1}) + \mathbf{f}_{\text{ext}}\Big), \\ \mathbf{x}^{n+1} &= \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\Big(-\nabla P(\mathbf{x}^{n+1/2}) - \nabla\Psi(\mathbf{x}^{n+1}) \\ &\qquad - \nabla\hat{C}_{\text{sf}}(\mathbf{x}^{n+1/2}) - \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}^{n+1}) - \nabla C_{\text{ss}}(\mathbf{x}^{n+1}) + \mathbf{f}_{\text{ext}}\Big).\end{aligned} \tag{B.4}$$

If we define

$$e(\mathbf{x}) = \left\|\mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\Big(-\nabla P(\mathbf{x}) - \nabla\Psi(\mathbf{x}) - \nabla C_{\text{sf}}(\mathbf{x}) - \nabla C_{\text{ss}}(\mathbf{x}) + \mathbf{f}_{\text{ext}}\Big) - \mathbf{x}\right\|, \tag{B.5}$$

$\mathbf{x}_{\mathrm{IE}}^{n+1}$ given by implicit Euler satisfies $e(\mathbf{x}_{\mathrm{IE}}^{n+1}) = 0$, while for $\mathbf{x}^{n+1}$ in our scheme, we have $e(\mathbf{x}^{n+1}) = \mathcal{O}(h^4)$. Specifically,

$$\begin{aligned}
e(\mathbf{x}^{n+1}) &= h^2 \Big\| \mathbf{M}^{-1}(\nabla P(\mathbf{x}^{n+1/2}) - \nabla P(\mathbf{x}^{n+1}) + \nabla \hat{C}_{\mathrm{sf}}(\mathbf{x}^{n+1/2}) - \frac{1}{2}\nabla C_{\mathrm{sf}}(\mathbf{x}^{n+1})) \Big\| \\
&= \mathcal{O}\Bigg(h^3 \Big\| \mathbf{M}^{-1} \Big(\nabla^2 P(\mathbf{x}^n)(\mathbf{v}^{n+1/2} - \mathbf{v}^{n+1}) + \frac{1}{2}\nabla^2 C_{\mathrm{sf}}(\mathbf{x}^n)(\mathbf{v}^{n+1/2} - \mathbf{v}^{n+1}) \Big) \Big\| \Bigg) \\
&= \mathcal{O}\Bigg(h^4 \Big\| \mathbf{M}^{-1} \Big(\nabla^2 P(\mathbf{x}^n) + \frac{1}{2}\nabla^2 C_{\mathrm{sf}}(\mathbf{x}^n) \Big) \mathbf{M}^{-1} \Big(\nabla \Psi(\mathbf{x}^n) + \frac{1}{2}\nabla C_{\mathrm{sf}}(\mathbf{x}^n) + \nabla C_{\mathrm{ss}}(\mathbf{x}^n) \Big) \Big\| \Bigg) \\
&= \mathcal{O}(h^4).
\end{aligned} \tag{B.6}$$

Here we assume that, in our discretized domain, the distance between any pair of primitives (particle-particle pair, particle-triangle pair, and triangle-triangle pair) has a lower bound ϵ. Thus, $\nabla^2 P(\mathbf{x}^n)$, $\nabla^2 C_{\mathrm{sf}}(\mathbf{x}^n)$, $\nabla \Psi(\mathbf{x}^n)$, $\nabla C_{\mathrm{sf}}(\mathbf{x}^n)$, and $\nabla C_{\mathrm{ss}}(\mathbf{x}^n)$ are all bounded, and this indicates that our method has an $\mathcal{O}(h^4)$ difference compared to the implicit Euler solution. Since implicit Euler has an $\mathcal{O}(h^2)$ error compared to the PDE solution, our proposed time splitting scheme shares the same order of accuracy with implicit Euler when it is stable.

Akinci, N., Cornelis, J., Akinci, G., and Teschner, M. (2013). Coupling elastic solids with smoothed particle hydrodynamics fluids. *Computer Animation and Virtual Worlds*, 24(3-4):195–203. 5, 37

Akinci, N., Ihmsen, M., Akinci, G., Solenthaler, B., and Teschner, M. (2012). Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (TOG)*, 31(4):1–8. 1, 5, 24, 37

Batty, C., Bertails, F., and Bridson, R. (2007). A fast variational framework for accurate solid-fluid coupling. *ACM Transactions on Graphics (TOG)*, 26(3):100–es. 4, 15

Batty, C., Uribe, A., Audoly, B., and Grinspun, E. (2012). Discrete viscous sheets. *ACM Transactions on Graphics (TOG)*, 31(4):1–7. 5

Becker, M., Ihmsen, M., and Teschner, M. (2009a). Corotated SPH for deformable solids. In *NPH*, pages 27–34. 6

Becker, M. and Teschner, M. (2007). Weakly compressible sph for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 209–217. 5, 16

Becker, M., Tessendorf, H., and Teschner, M. (2009b). Direct forcing for Lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):493–503. 5, 24

Bender, J. and Koschier, D. (2015). Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 147–155. 5, 16, 36, 37

Bender, J. and Koschier, D. (2016). Divergence-free SPH for incompressible and viscous fluids. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1193–1206. 18

Bender, J., Kugelstadt, T., Weiler, M., and Koschier, D. (2019). Volume maps: An implicit boundary representation for sph. In *Motion, Interaction and Games*, pages 1–10. 6, 37

Bonet, J. and Lok, T.-S. (1999). Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. *Computer Methods in Applied Mechanics and Engineering*, 180(1-2):97–115. 6

Brandt, C., Scandolo, L., Eisemann, E., and Hildebrandt, K. (2019). The reduced immersed method for real-time fluid-elastic solid interaction and contact simulation. *ACM Transactions on Graphics (TOG)*, 38(6):1–16. 5

Bridson, R. (2015). *Fluid Simulation for Computer Graphics*. AK Peters/CRC Press. 15

Brookshaw, L. (1985). A method of calculating radiative heat diffusion in particle simulations. *Publications of the Astronomical Society of Australia*, 6(2):207–210. 13

Chen, Y., Davis, T. A., Hager, W. W., and Rajamanickam, S. (2008). Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)*, 35(3):1–14. 28

Clausen, P., Wicke, M., Shewchuk, J. R., and O'brien, J. F. (2013). Simulating liquids and solid-liquid interactions with Lagrangian meshes. *ACM Transactions on Graphics (TOG)*, 32(2):1–15. 5

Fang, Y., Qu, Z., Li, M., Zhang, X., Zhu, Y., Aanjaneya, M., and Jiang, C. (2020). IQ-MPM: An interface quadrature material point method for non-sticky strongly two-way coupled nonlinear solids and fluids. *ACM Transactions on Graphics (TOG)*, 39(4):51–1. 1, 7

Fedkiw, R. P. (2002). Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *Journal of Computational Physics*, 175(1):200–224. 4

Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S. (1999). A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152(2):457–492. 4

Fei, Y., Batty, C., Grinspun, E., and Zheng, C. (2018). A multi-scale model for simulating liquid-fabric interactions. *ACM Transactions on Graphics (TOG)*, 37(4):1–16. 6

Ganzenmüller, G. C. (2015). An hourglass control algorithm for lagrangian smooth particle hydrodynamics. *Computer Methods in Applied Mechanics and Engineering*, 286:87–106. 6

Gao, M., Pradhana, A., Han, X., Guo, Q., Kot, G., Sifakis, E., and Jiang, C. (2018). Animating fluid sediment mixture in particle-laden flows. *ACM Transactions on Graphics (TOG)*, 37(4):1–11. 6

Gissler, C., Peer, A., Band, S., Bender, J., and Teschner, M. (2019). Interlinked SPH pressure solvers for strong fluid-rigid coupling. *ACM Transactions on Graphics (TOG)*, 38(1):1–13. 1, 5

Guendelman, E., Selle, A., Losasso, F., and Fedkiw, R. (2005). Coupling water and smoke to thin deformable and rigid shells. *ACM Transactions on Graphics (TOG)*, 24(3):973–981. 4

Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., and Jiang, C. (2018). A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4):1–14. 7

Hyde, D. A., Gagniere, S. W., Marquez-Razon, A., and Teran, J. (2020). An implicit updated Lagrangian formulation for liquids with large surface energy. *ACM Transactions on Graphics (TOG)*, 39(6):1–13. 17

Ihmsen, M., Akinci, N., Gissler, M., and Teschner, M. (2010). Boundary handling and adaptive time-stepping for PCISPH. In *VRIPHYS'10*, pages 79–88. 5, 24

Ihmsen, M., Cornelis, J., Solenthaler, B., Horvath, C., and Teschner, M. (2013). Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):426–435. 5, 16, 36, 37

Jiang, C., Schroeder, C., Teran, J., Stomakhin, A., and Selle, A. (2016). The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, pages 1–52. 1, 6

Klingner, B. M., Feldman, B. E., Chentanez, N., and O'brien, J. F. (2006). Fluid animation with dynamic meshes. In *ACM SIGGRAPH 2006 Papers*, pages 820–825. 4, 5

Koschier, D. and Bender, J. (2017). Density maps for improved SPH boundary handling. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–10. 6, 37

Koschier, D., Bender, J., Solenthaler, B., and Teschner, M. (2022). A survey on SPH methods in computer graphics. *Computer Graphics Forum*, 41(2):737–760. 5

Kugelstadt, T., Bender, J., Fernández-Fernández, J. A., Jeske, S. R., Löschner, F., and Longva, A. (2021). Fast corotated elastic SPH solids with implicit zero-energy mode control. *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques*, 4(3):1–21. 6, 37, 43

Levin, D. I., Litven, J., Jones, G. L., Sueda, S., and Pai, D. K. (2011). Eulerian solid simulation with contact. *ACM Transactions on Graphics (TOG)*, 30(4):1–10. 4

Li, M., Ferguson, Z., Schneider, T., Langlois, T. R., Zorin, D., Panozzo, D., Jiang, C., and Kaufman, D. M. (2020). Incremental potential contact: Intersection-and inversion-free, large-deformation dynamics. *ACM Trans. Graph.*, 39(4):49. 1, 7, 16, 20, 21, 23, 37

Li, X., Fang, Y., Li, M., and Jiang, C. (2022). BFEMP: Interpenetration-free mpm-fem coupling with barrier contact. *Computer Methods in Applied Mechanics and Engineering*, 390:114350. 7

Macklin, M. and Müller, M. (2013). Position based fluids. *ACM Transactions on Graphics (TOG)*, 32(4):1–12. 1, 16

Monaghan, J. J. (1992). Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30:543–574. 18

Monaghan, J. J. (1994). Simulating free surface flows with SPH. *Journal of computational physics*, 110(2):399–406. 5

Monaghan, J. J. (2005). Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68(8):1703. 18

Moreau, J. J. (2011). On unilateral constraints, friction and plasticity. In *New Variational Techniques in Mathematical Physics*, pages 171–322. Springer. 16

Müller, M., Charypar, D., and Gross, M. H. (2003). Particle-based fluid simulation for interactive applications. In *Symposium on Computer Animation*, volume 2. 18

Peer, A., Gissler, C., Band, S., and Teschner, M. (2018). An implicit SPH formulation for incompressible linearly elastic solids. *Computer Graphics Forum*, 37(6):135–148. 6, 37, 43

Peer, A., Ihmsen, M., Cornelis, J., and Teschner, M. (2015). An implicit viscosity formulation for SPH fluids. *ACM Transactions on Graphics (TOG)*, 34(4):1–10. 18

Peer, A. and Teschner, M. (2016). Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion. *IEEE Transactions on Visualization and Computer Graphics*, 23(12):2656–2662. 18

Peskin, C. S. (2002). The immersed boundary method. *Acta Numerica*, 11:479–517. 5

Robinson-Mosher, A., Schroeder, C., and Fedkiw, R. (2011). A symmetric positive definite formulation for monolithic fluid structure interaction. *Journal of Computational Physics*, 230(4):1547–1566. 4

Robinson-Mosher, A., Shinar, T., Gretarsson, J., Su, J., and Fedkiw, R. (2008). Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Transactions on Graphics (TOG)*, 27(3):1–9. 4

Roble, D., Zafar, N. b., and Falt, H. (2005). Cartesian grid fluid simulation with irregular boundary voxels. In *ACM SIGGRAPH 2005 Sketches*, pages 138–es. 4

Sifakis, E. and Barbic, J. (2012). Fem simulation of 3D deformable solids: A practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 courses*, pages 1–50. 16

Solenthaler, B. and Pajarola, R. (2009). Predictive-corrective incompressible SPH. In *ACM SIGGRAPH 2009 papers*, pages 1–6. 5

Solenthaler, B., Schläfli, J., and Pajarola, R. (2007). A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds*, 18(1):69–82. 6

Stomakhin, A., Howes, R., Schroeder, C. A., and Teran, J. M. (2012). Energetically consistent invertible elasticity. In *Symposium on Computer Animation*, pages 25–32. 27

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., and Selle, A. (2013). A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4):1–10. 6

Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., and Selle, A. (2014). Augmented MPM for phase-change and varied materials. *ACM Transactions on Graphics (TOG)*, 33(4):1–11. 6

Sulsky, D., Zhou, S.-J., and Schreyer, H. L. (1995). Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87(1-2):236–252. 6

Takahashi, T. and Batty, C. (2020). Monolith: A monolithic pressure-viscosity-contact solver for strong two-way rigid-rigid rigid-fluid coupling. *ACM Transactions on Graphics*, 39(6):1–16. 4

Takahashi, T. and Batty, C. (2021). FrictionalMonolith: A monolithic optimization-based approach for granular flow with contact-aware rigid-body coupling. *ACM Transactions on Graphics (TOG)*, 40(6):1–20. 4

Takahashi, T. and Batty, C. (2022). ElastoMonolith: A monolithic optimization-based liquid solver for contact-aware elastic-solid coupling. *ACM Transactions on Graphics (TOG)*, 41(6):1–19. 30, 36, 37, 38, 43

Takahashi, T., Dobashi, Y., Fujishiro, I., Nishita, T., and Lin, M. C. (2015). Implicit formulation for SPH-based viscous fluids. *Computer Graphics Forum*, 34(2):493–502. 18

Tampubolon, A. P., Gast, T., Klár, G., Fu, C., Teran, J., Jiang, C., and Museth, K. (2017). Multi-species simulation of porous sand and water mixtures. *ACM Transactions on Graphics (TOG)*, 36(4):1–11. 6

Teng, Y., Levin, D. I., and Kim, T. (2016). Eulerian solid-fluid coupling. *ACM Transactions on Graphics (TOG)*, 35(6):1–8. 1, 5

Valkov, B., Rycroft, C. H., and Kamrin, K. (2015). Eulerian method for multiphase interactions of soft solid bodies in fluids. *Journal of Applied Mechanics*, 82(4):041011. 1, 5

Wang, H., Jin, Y., Luo, A., Yang, X., and Zhu, B. (2020). Codimensional surface tension flow using moving-least-squares particles. *ACM Transactions on Graphics (TOG)*, 39(4):42–1. 5

Weiler, M., Koschier, D., Brand, M., and Bender, J. (2018). A physically consistent implicit viscosity solver for SPH fluids. *Computer Graphics Forum*, 37(2):145–155. 18

Yan, X., Li, C.-F., Chen, X.-S., and Hu, S.-M. (2018). MPM simulation of interacting fluids and solids. *Computer Graphics Forum*, 37(8):183–193. 6

Zarifi, O. and Batty, C. (2017). A positive-definite cut-cell method for strong two-way coupling between fluids and deformable bodies. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–11. 1, 4, 39, 43

Zhang, F. (2006). *The Schur Complement and its Applications*, volume 4. Springer Science & Business Media. 28