UNIVERSITY OF CALIFORNIA
Los Angeles

Particle Grid Hybrid Methods for Multi-Material Dynamics

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Aerospace Engineering

by

Alan Marquez Razon

2021

ABSTRACT OF THE DISSERTATION

Particle Grid Hybrid Methods for Multi-Material Dynamics

by

Alan Marquez Razon
Doctor of Philosophy in Aerospace Engineering
University of California, Los Angeles, 2021
Professor Jeffrey Eldredge, Co-Chair
Professor Joseph M Teran, Co-Chair

The Material Point Method (MPM) has shown its capability in simulating multi-physics and multi-material dynamics. In this dissertation, we present an extension to the Material Point Method (MPM) for simulating volumetric collisions of elastic objects, formulate a novel approach for surface tension phenomenon, and a hybrid particle/grid approach for simulating incompressible fluids. First, we present a momentum-conserving hybrid particle/grid iteration method for volumetric elastic contact problems. We use a Lagrangian mesh to discretize the elastic material and an Eulerian grid to provide the collision response at the mesh boundary. The Eulerian grid utilizes Affine-Particle-In-Cell (APIC) transfers which conserve both linear and angular momentum when affine information is provided to the boundary nodes. The transfers are leveraged in terms of performance to capture the impulses needed to prevent a collision. The cohesion that occurs when separating in the APIC transfers is removed through augury iterations. A novel resampling scheme and modifications to the transfers that conserve mass, linear and angular momentum are used to capture the collision on finner Eulerian grids. This iteration can be used to provide faster convergence

in impulse-based approaches that are commonly used in graphics. Second, we present an updated Lagrangian discretization of surface tension forces for the simulation of liquids with moderate to extreme surface tension effects. The potential energy associated with surface tension is proportional to the surface area of the liquid. We design discrete forces as gradients of this energy to the motion of the fluid over a time step. We show that this naturally allows for inversion of the Hessian of the potential energy required with the use of Newton's method to solve the systems of nonlinear equations associated with implicit time stepping. We design a novel level-set-based boundary quadrature technique to discretize the surface area calculation in our energy-based formulation. Our approach works most naturally with Particle-In-Cell techniques and we demonstrate our approach with a weakly incompressible model for liquid discretized with the Material Point Method. Lastly, we design a particle resampling approach needed to achieve perfect conservation of linear and angular momentum with APIC. We show that our approach is essential for allowing efficient implicit numerical integration in the limit of materials with variable high surface tension. Last, we present a hybrid particle/grid approach for simulating incompressible fluids on collocated velocity grids. We interchangeably use particle and grid representations of transported quantities to balance efficiency and accuracy. A novel Backward Semi-Lagrangian method is derived to improve the accuracy of grid-based advection. Our approach utilizes the implicit formula associated with solutions of Burgers' equation. We enforce incompressibility over collocated, rather than staggered grids. Our projection technique is variational and designed for B-spline interpolation over regular grids where multi quadratic interpolation is used for velocity and multilinear interpolation for pressure. Despite our use of regular grids, we extend the variational technique to allow for cut-cell definition of irregular flow domains for both Dirichlet and free surface boundary conditions.

The dissertation of Alan Marquez Razon is approved.

Kunihiko Taira

Mohammad Khalid Jawed

Joseph M Teran, Committee Co-Chair

Jeffrey Eldredge, Committee Co-Chair

University of California, Los Angeles

2021

To my family, friends, professors, and mentors
that continue to support me with their advice and teachings

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank my current advisor Joseph Teran for his constant support and mentorship throughout the later years of the doctoral program. His enthusiasm for pursuing a topic has proven to be an exemplar in striving for research.

I would like to thank Professor Jeff Eldredge for his support through uncertain times and advice to keep moving forward.

I would also like to thank Professor Mohammad Khalid Jawed and Professor Kunihiko Taira for accepting to be part of the committee and providing an exemplary role model to follow.

I would like to thank James Lewis for the internship opportunity and for broadening the perspective of what a researcher can become.

I would like to thank Michael Tupek and Jacob Koester for the opportunity to work along with them and their support.

I would like to thank Oddvar Bendiksen for his mentorship and support during the initial years of the doctoral program.

I would like to thank Luciano Demasi for his continued support as a mentor and unique perspective.

Kindest regards to the Teran Lab members: David Hyde, Steven Gagniere, Ayano Kaneda, Jingyu Chen, Victoria Kala, Elias Gueidon, Qi Guo, Stephanie Wang, Xuchen Han, Yizhou Chen, and Yushan Han that have helped me grow into a better person and researcher.

I would like to acknowledge the encouragement provided by my friends from LA, SD and Mexico.

Lastly, I would like to thank my immediate family Elba Razon, Heriberto Marquez, Grisel Marquez-Razon, and Kevin Marquez-Razon, and the rest of my family for their constant love and support.

VITA

2013–2015 M.S. (Aerospace Engineering), UCLA, Los Angeles, California.

2008–2013 B.S. (Aerospace Engineering), SDSU, San Diego, California.

2015-2016 Aerospace Engineering Intern, Jet Propulsion Laboratory, Mars Oxygen IRSU Experiment (MOXIE), Pasadena, California.

2020-2021 Graduate Student Researcher, Mathematics Department, UCLA, California.

2016-2017 Graduate Student Researcher, Mechanical and Aerospace Engineering Department, UCLA, California.

2015-2019 Teaching Assistant, Mechanical and Aerospace Engineering Department, "Flight Mechanics and Preliminary Aircraft Designs," UCLA, California.

2016 Teaching Assistant, Mechanical and Aerospace Engineering Department, "Aeroelastic effects on structures," UCLA, California.

PUBLICATIONS

"A Momentum-Conserving Implicit Material Point Method for Surface Energies with Spatial Gradients." J Chen, V Kala, A Marquez-Razon, E Gueidon, DAB Hyde, J Teran. ACM TOG (SIGGRAPH 2021 Technical Papers), 2021, DOI: 10.1145/3450626.3459874

"An implicit updated lagrangian formulation for liquids with large surface energy." David Hyde, Steven Gagniere, Alan Marquez-Razon, Joseph Teran ACM TOG (SIGGRAPH Asia

2020 Technical Papers), 2020, DOI: 10.1145/3414685.3417845

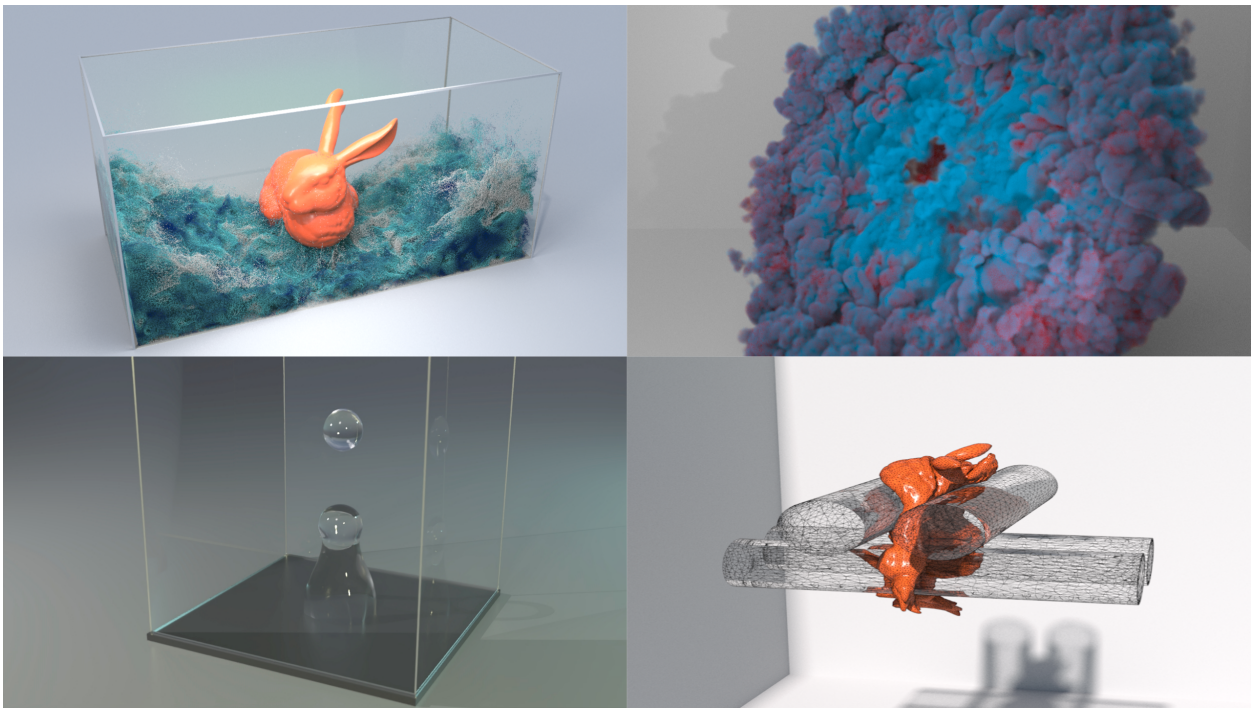
"A Hybrid Lagrangian/Eulerian Collocated Advection and Projection Method for Fluid Simulation." Steven Gagniere, David Hyde, Alan Marquez-Razon, Chenfanfu Jiang, Ziheng Ge, Xuchen Han, Qi Guo, Joseph Teran, Symposium on Computer Animation (SCA), 2020, DOI: 10.1111/cgf.14096

"Phenomenology of nonlinear aeroelastic responses of highly deformable joined wings." Rauno Cavallaro, Andrea Iannelli, Luciano Demasi and Alan Marquez Razon. Advances in Aircraft and Spacecraft Science, An International Journal 2015. DOI: 10.12989/aas.2015.2.2.125

"Phenomenology of Nonlinear Aeroelastic Responses of Highly Deformable Joined-wings Configurations." Rauno Cavallaro, Andrea Iannelli, Luciano Demasi and Alan Marquez Razon, AIAA Journal 2014. https://doi.org/10.2514/6.2014-1199

CHAPTER 1

Introduction



Accurate simulation of interactions between multiple materials continues to be challenging for a wide variety of applications in engineering and computational graphics. Modeling the momentum transfer that occurs in the interface between continuums is crucial to capture the physical phenomena of the simulations. The material point method (MPM) [1], which derives from the particle in cell (PIC)[2] method, can implicitly capture the interfaces' interactions. In addition, MPM can be used to model the response of multiple materials, resolve the collisions between the materials, and incorporate multiphysics relationships. Regardless of the advantages, MPM possesses drawbacks that narrow its application in engineering ap-

plications. One drawback that affects the dynamics of the simulation is the dissipation that damps angular momentum and energy. Improvements in MPM have achieved conservation of angular momentum [3] and better energy conservation [4]. Another drawback is the loss of information on the transfer where particles distributions are too sparse to provide sufficiently accurate quadrature [3]. We will show that resampling of the particles preserves the conservation of the system while maintaining the overall dynamics of the system. Another drawback is that the transfers can cause numerical cohesion that is resolution-dependent. While this cohesion decreases with the resolution, an augury filter [5] can be used to remove unwanted cohesion in the collision that is linear and angular momentum conserving.

This dissertation's focus is improving the current MPM implementation and overcoming drawbacks, for simulations involving hybrid MPM/FEM for volumetric collision, modeling surface tension with MPM, and hybrid Lagrangian/Eulerian methods for incompressible fluids.

1.1 Dissertation Overview

The dissertation is structured as follows:

Chapter 2: Background Material Background information on kinematics, continuum mechanics, and elasticity are provided.

Chapter 3: Material Point Method The material point method space discretization, transfers, and time discretization for explicit forward Euler and implicit backward Euler are provided. This chapter is the baseline of the methods in the dissertation.

Chapter 4: Hybrid Particle/Grid Iteration for Volumetric Elastic Contact The finite element formulation is provided. The material point method is used to couple self-collision and collision between different volumetric materials modeled with finite elements. The method explains the resampling scheme used to support finer Eulerian grids while conserving linear and angular momentum. The augury iterations are presented and used to

remove the cohesion when separating inherited from MPM.

Chapter 5: Implicit Surface Tension in MPM Surface tension is incorporated into the material point method derived from a strain energy density function. The level set approach to discretize the surface is presented. Particle boundary resampling is introduced to avoid errors in the surface tension forces discretization. The implicit formulation is presented to support materials with large surface tension and large bulk modulus.

Chapter 6: Hybrid Lagrangian/Eulerian Collocated Advection and Projection Method We present a hybrid particle/grid approach for simulating incompressible fluids on collocated velocity grids. The particle (MPM) and grid (FEM) representation of transported quantities is interchangeably used to balance efficiency and accuracy. Despite our use of regular grids, we extend the variational technique to allow for cut-cell definition of irregular flow domains for both Dirichlet and free surface boundary conditions.

CHAPTER 2

Background Material

2.1 Kinematics

Kinematics describes the motion that occurs in the continuum. This motion includes translations, rotations, and deformations. In a simulation, having different frames of reference can be advantageous. Primarily, the material point method uses two reference frames. The Lagrangian frame of reference measures the kinematics from the original position. The Eulerian frame of reference measures the kinematics from a fixed position at the current time.

The current postion $\mathbf{x}$ can be referenced to the initial position $\mathbf{X}$ with a map $\boldsymbol{\phi}$.

$$\mathbf{x}(t) = \boldsymbol{\phi}(\mathbf{X}, t) \tag{2.1}$$

The velocity and acceleration are calculated with the time derivatives:

$$\begin{aligned} \mathbf{V}(t) &= \tfrac{\partial \boldsymbol{\phi}(\mathbf{X},t)}{\partial t} \\ \mathbf{A}(t) &= \tfrac{\partial \boldsymbol{V}(\mathbf{X},t)}{\partial t} \end{aligned} \tag{2.2}$$

The Eulerian view can be obtained with the inverse map as follows:

$$\begin{aligned} \mathbf{v}(\mathbf{x}, t) &= \mathbf{V}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t) \\ \mathbf{a}(\mathbf{x}, t) &= \mathbf{A}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t) \end{aligned} \tag{2.3}$$

Similarly the Lagrangian view becomes:

$$\begin{aligned}\mathbf{V}(\mathbf{X},t) &= \mathbf{v}(\boldsymbol{\phi}(\mathbf{X},t),t)\\ \mathbf{A}(\mathbf{X},t) &= \mathbf{a}(\boldsymbol{\phi}(\mathbf{X},t),t)\end{aligned} \tag{2.4}$$

The local deformation of the material is represented by the Jacobian of the map function and is refered as the deformation gradient $\mathbf{F}$.

$$\begin{aligned}\mathbf{F}(\mathbf{X},t) &= \tfrac{\partial \mathbf{x}}{\partial \mathbf{X}}(\mathbf{X},t)\\ J &= \det(\mathbf{F})\end{aligned} \tag{2.5}$$

2.2 Elasticity

Our methods are moddeled with elastic materials, thsi means that the deformation is perferctly reversible. The stress-strain relationships for deformable materials are expressed with a strain energy density function $\Psi(\mathbf{F})$, the deformation gradient $\mathbf{F}$, and the first Piola-Kirchoff Stress tensor $\mathbf{P}$.

$$\mathbf{P}(\mathbf{X},t) = \frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{X},t) \tag{2.6}$$

Furthermore, the Cauchy stress $\boldsymbol{\sigma}$ can be related to the first Piola-Kirchoff Stress tensor.

$$\boldsymbol{\sigma} = \frac{1}{J}\mathbf{P}\mathbf{F}^T \tag{2.7}$$

These relationships allow us to model widely used materials, as long as the strain energy density function is defined. The materials that are used to model the solid continuums consist of Linear elasticity, Neo-Hookean elasticity, and Corotated Elasticity. The material used to model fluids is the compressible flow model.

Strain can be measured from the deformation gradient with the Green strain tensor $\mathbf{E}$

$$\mathbf{E} = \frac{1}{2}(\mathbf{F}^T\mathbf{F} - \mathbf{I}) \tag{2.8}$$

2.2.1 Linear Elasticity

The linear elastic model works only when handling small deformations. The formulation of linear elasticity comes by approximating the Green strain tensor with an infinitesimal strain tensor $\boldsymbol{\epsilon}$. This approximation has two main consequences; stress and strain are related linearly requiring lower computations, and a rigid rotation motion on $\mathbf{F}$ can generates non zero strain $\boldsymbol{\epsilon} = \frac{1}{2}(\mathbf{R} + \mathbf{R}^T) - \mathbf{I}$ resulting in deformation. We show the material strain energy and Piola stress in terms of the Lame coefficients μ and λ.

$$\boldsymbol{\epsilon} = \frac{1}{2}(\mathbf{F}^T + \mathbf{F}) - \mathbf{I} \tag{2.9}$$

$$\Psi = \mu\boldsymbol{\epsilon} : \boldsymbol{\epsilon} + \frac{\lambda}{2}\mathbf{tr}^2(\boldsymbol{\epsilon}) \tag{2.10}$$

$$\mathbf{P} = 2\mu\boldsymbol{\epsilon} + \lambda\mathbf{tr}(\boldsymbol{\epsilon})\mathbf{I} \tag{2.11}$$

2.2.2 Neo-Hookean Elasticity

The strain energy density function for the Neo-Hookean describes isotropic elastic materials. This strain energy density function Ψ can be expressed in terms of the deformation gradient $\mathbf{F}$, the Lame parameters μ and λ, the Jacobean $J = |\mathbf{F}|$, and d being the trace of the identity matrix.

$$\Psi(\mathbf{F}) = \frac{\mu}{2}(F_{ij}F_{ij} - d) - \mu \ln J + \frac{\lambda}{2}(\ln J)^2 \tag{2.12}$$

The first Piola-Kirchhoff stress $\mathbf{P}$ is defined as the derivative of the strain energy density function to the deformation gradient.

$$\mathbf{P} = \mu\mathbf{F} + (\lambda \ln J - \mu)\mathbf{F}^{-T} \tag{2.13}$$

The material has strong resistance to compression with the strain energy density ap-

proaching infinity as the Jacobian approaches zero. Without modifications, the model will not support deformations close to zero or inversion.

2.2.3 Corrected Corotated Elasticity

Corotated linear elasticity is a constitutive model that attempts to combine the simplicity of the stress-deformation relationship in a linear material with enough nonlinear characteristics to secure rotational invariance. We use the Elasticity model described in [6].

$$\Psi = \mu \sum_{\alpha=0}^{d-1} (\sigma_\alpha - 1)^2 + \frac{\lambda}{2}(J-1)^2 \tag{2.14}$$

$$\sigma_\alpha = \mathbf{\Sigma}_{\alpha\alpha} \tag{2.15}$$

$$\mathbf{F} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.16}$$

$$\mathbf{P} = 2\mu(\mathbf{F} - \mathbf{U}\mathbf{V}^T) + \lambda(J-1)\mathbf{F}^{-T} \tag{2.17}$$

2.2.4 Compressible Flow Model

The compressible flow model utilizes is derived from pressure with the material deformation being dependent in the Jacobian J and bulk modulus of the liquid λ.

$$\Psi(\mathbf{J}) = \frac{\lambda}{2}(J-1)^2 \tag{2.18}$$

$$\mathbf{P} = \lambda(J-1)\mathbf{F}^{-T} \tag{2.19}$$

2.3 Governing Equations

The governing equations in the Eulerian view for continuum mechanics are the conservation of mass and conservation of linear momentum where ρ is the density, $\mathbf{v}$ is the velocity, $\mathbf{g}$ is

the gravity vector, and $\boldsymbol{\sigma}$ is the Cauchy stress tensor.

$$\begin{aligned} \tfrac{D\rho}{Dt} &= -\rho \nabla \cdot \mathbf{v} \\ \rho \tfrac{D\mathbf{v}}{Dt} &= \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} \end{aligned} \tag{2.20}$$

In the Lagrangian view, the governing equations take the following form where $R(\mathbf{X}, t)$ is the mass density at point $\mathbf{X} \in \Omega^0$, $\mathbf{V}$ is the velocity in Lagrangian view, and $\mathbf{P}$ the first Piola stress.

$$\begin{aligned} R(\mathbf{X}, t) J(\mathbf{X}, t) &= R(\mathbf{X}, 0) \\ R(\mathbf{X}, 0) \tfrac{\partial \mathbf{V}}{\partial t} &= \nabla \cdot \mathbf{P} + R(\mathbf{X}, 0) \mathbf{g} \end{aligned} \tag{2.21}$$

CHAPTER 3

The Material Point Method

Interactions between multiple materials are nonlinear initial boundary value problems, where the initial conditions and boundary conditions will influence the resulting behavior. Accurate synchronized modeling of the interface is needed to describe the behavior of their interaction. One hybrid particle-grid method capable of handling the interactions without additional calculations to define the interface is MPM [1]. MPM utilizes two frames of reference, an Eulerian and a Lagrangian. The material points are defined in the Lagrangian reference frame and that are used to discretize the continuum. The material particles store the current state of the continuum. The state transferred to the background Eulerian reference frame. Here the momentum is updated, advanced in time, and transferred back to the material particles. The method leverages both reference frames to avoid entanglement commonly seen in Lagrangian methods and excessive dissipation from Eulerian methods. In addition, the interface boundary along with the non-slip condition is captured by the momentum transfers [7]. The B-splines compact stencils used in MPM can improve the computational performance with friendly parallelization.

3.1 Discrete Form

We discretize the weak form of the equations by multiplying by a test function $w = w_i N_i$ where w_i is arbitrary and N_i is the interpolation shape function. Where it is zero on the prescribed sections of the domain. Furthermore, our discretization for the mass and momentum will be:

$$
\begin{aligned}
&M_{ij} = \sum_{p=1}^{N_p} m_p N_i(\mathbf{x}_p(t)) N_j(\mathbf{x}_p(t)) \\
&\sum_{j=1}^{N_n} M_{ij}(t) \tfrac{d\mathbf{v}_j}{dt} = \mathbf{f}_i^{int} + \mathbf{f}_i^{ext}
\end{aligned}
\tag{3.1}
$$

The external forces can be discretized in terms of traction and body forces:

$$
\begin{aligned}
&\mathbf{f}_i^{ext} = \mathbf{b}_i(t) + \hat{\mathbf{t}}_i(t) \\
&\hat{\mathbf{t}}_i(t) = \textstyle\int_{\partial\Omega} \mathbf{t}(\mathbf{x}, t) N_i(\mathbf{x}_p(t)) dS \\
&\mathbf{b}_i(t) = \textstyle\sum_{p=1}^{N_p} m_p \mathbf{b}(\mathbf{x}_p(t), t) N_i(\mathbf{x}_p(t))
\end{aligned}
\tag{3.2}
$$

The internal forces can be discretized as:

$$
\mathbf{f}_i^{int} = -\sum_{p=1}^{N_p} V_p(t) \boldsymbol{\sigma}_p(t) \nabla N_i(\mathbf{x}_p(t)) \tag{3.3}
$$

3.2 B-spline Interpolation

Quadratic B-splines have the following properties that make them suitable as interpolating functions: 2nd order, a more compact stencil compared to higher-order B-splines and can get comparable results to cubic B-splines. One drawback is that extra interpolation steps are needed to avoid spontaneous growth when interpolating oscillatory data. Other interpolating functions can also be suitable, but B-splines compactness couples well with MPM. The quadratic B-splines selected are described below with values depending on the particle position $\mathbf{x}_p$, the background grid position $\mathbf{x}_i$, and the background cartesian grid spacing dx.

$$
N\left(\frac{x_p - x_i}{dx}\right) = N(x) = \begin{Bmatrix} \frac{3}{4} - |x|^2 & 0 \le |x| \le \frac{1}{2} \\ \frac{1}{2}(\frac{3}{2} - |x|)^2 & \frac{1}{2} \le |x| \le \frac{3}{2} \\ 0 & \frac{3}{2} \le |x| \end{Bmatrix}
$$

3.3 Momentum Transfers

PIC method is dissipative and reduces the angular momentum. FLIP transfer can conserve angular momentum, but it is noisy and is affected by the zero modes. This noise can lead to

unstable behavior. APIC can preserve the angular momentum while being smoother than FLIP. However, slight dissipation can still be observed in APIC when using mass lumping. APIC transfers are shown below [8].

Interpolation weight:

$$w_{ip}^n = \mathbf{N}(\frac{\mathbf{x}_p - \mathbf{x}_i}{dx}) = N(\frac{x_p - x_i}{dx})N(\frac{y_p - y_i}{dx})N(\frac{z_p - z_i}{dx}) \tag{3.4}$$

Conservation of the grid mass:

$$m_i^n = \sum_p m_p w_{ip}^n \tag{3.5}$$

Conservation of the grid momentum:

$$m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{A}_p^n(\mathbf{x}_i - \mathbf{x}_p^n)) \tag{3.6}$$

Internal forces:

$$\mathbf{f}_i^n = \mathbf{f}_i(\mathbf{x}^n) = -\sum_p V_p^n \boldsymbol{\sigma}_p^n \nabla w_{ip}^n \tag{3.7}$$

We solve for the new grid velocities where the internal and external forces are evaluated at t^n for explicit time integration and t^{n+1} for implicit time integration.

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n + \Delta t \frac{\mathbf{f}_i^{int+ext}}{m_i^n} \tag{3.8}$$

The rest of the particle variables can be updated.

$$\begin{aligned} \mathbf{v}_p^{n+1} &= \textstyle\sum_p w_{ip}^n \mathbf{v}_i^{n+1} \\ \mathbf{A}_p^{n+1} &= \tfrac{4}{\Delta x^2} \textstyle\sum_i w_{ip} \mathbf{v}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^T \\ \mathbf{x}_p^{n+1} &= \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1} \end{aligned} \tag{3.9}$$

3.3.1 Explicit Time Discretization

The update of momentum requires that the force is evaluated at t^n for the explicit MPM. The discretize momentum update becomes:

$$\sum_{j=1}^{N_n} M_{ij}(t^n)\frac{\mathbf{v}_j(t^{n+1}) - \mathbf{v}_j(t^n)}{\Delta t} = \mathbf{f}_i(x^n) \tag{3.10}$$

3.3.2 Implicit Discretization

The implicit momentum update requires the forces at the same time as the update. With backward Euler discretization, the conservation of momentum implies that the forces are implicitly dependent on the grid motion.

$$\sum_{j=1}^{N_n} M_{ij}(t^n)\frac{\hat{\mathbf{v}}_j(t^{n+1}) - \mathbf{v}_j(t^n)}{\Delta t} = \mathbf{f}_i(x^{n+1}) = \mathbf{f}_i(x^n + \Delta t \mathbf{v}_j(t^{n+1})) \tag{3.11}$$

with boundary conditions satisfied by

$$\hat{\mathbf{v}}^{n+1} = \mathbf{Z}\tilde{\mathbf{v}}^{n+1}. \tag{3.12}$$

The elastic force $f_{i\alpha}$ is defined as

$$f_{i\alpha}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial \Psi}{\partial F_{\alpha\gamma}}\left(\mathbf{F}_p\left(\hat{\mathbf{x}}\right)\right) F^n_{p\delta\gamma}\frac{\partial N_{\mathbf{i}}}{\partial x_\delta}(\mathbf{x}^n_p)V^n_p \tag{3.13}$$

Solving the systems for $\tilde{\mathbf{v}}^{n+1}$ with the Newton's method.

3.3.3 Residual

For this section $g_i(\tilde{\mathbf{v}})$ is used for residual of the nonlinear equations

$$g_i(\tilde{\mathbf{v}}) = Z_{\mathbf{i}\alpha i}\left(m^n_{\mathbf{i}}\frac{Z_{\mathbf{i}\alpha k}\tilde{v}_k - v^n_{\mathbf{i}\alpha}}{\Delta t} - f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t\mathbf{Z}\tilde{\mathbf{v}}\right) - m^n_{\mathbf{i}}g_\alpha\right) \tag{3.14}$$

where in this case summation on $\mathbf{i}$ and α is implied.

3.3.4 Linearization

The linearization of the residual has the form

$$\frac{\partial g_i}{\partial \tilde{v}_j}(\tilde{\mathbf{v}}) = Z_{\mathbf{i}\alpha i}\left(\frac{m^n_{\mathbf{i}}}{\Delta t}Z_{\mathbf{i}\alpha j} - \Delta t\frac{\partial f_{\mathbf{i}\alpha}}{\partial \hat{x}_{\mathbf{j}\beta}}\left(\mathbf{x} + \Delta t\mathbf{Z}\tilde{\mathbf{v}}\right)Z_{\mathbf{j}\beta j}\right) \tag{3.15}$$

where the linearization of the elastic force satisfies Equation (3.16)

$$\frac{\partial f_{\mathbf{i}\alpha}}{\partial \hat{x}_{\mathbf{j}\beta}}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial^2 \Psi}{\partial F_{\beta\epsilon} F_{\alpha\gamma}} \left(\mathbf{F}_p\left(\hat{\mathbf{x}}\right)\right) F^n_{p\delta\gamma} F^n_{p\sigma\epsilon} \frac{\partial N_{\mathbf{i}}}{\partial x_\delta}(\mathbf{x}^n_p) \frac{\partial N_{\mathbf{j}}}{\partial x_\sigma}(\mathbf{x}^n_p) V^n_p \tag{3.16}$$

.

3.3.5 Implementation

The formula to update the approximation with Newton's method is

$$\begin{aligned} &\tfrac{\partial g_i}{\partial \tilde{v}_j}(\mathbf{w}^k)\delta w^k_j = -g_i(\mathbf{w}^k) \\ &\mathbf{w}^{k+1} = \mathbf{w}^k + \delta \mathbf{w}^k \end{aligned} \tag{3.17}$$

(3.18)

where $\mathbf{w}^0$ is the initial guess to $\tilde{\mathbf{v}}^{n+1}$ and $\mathbf{w}^k \to \tilde{\mathbf{v}}^{n+1}$.

3.3.6 Newton Residual

The right-hand side of the system for the Newton increment $\delta \mathbf{w}^k$ is computed. We call this computation of the "Newton Residual" $-g_i(\mathbf{w}^k)$:

1. Transfer grid values to particles to compute $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)$ using current guess to grid velocity $\mathbf{w}^k$.

2. Use $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)$ to update $\mathbf{F}_p$:

$$\mathbf{F}_p = \hat{\mathbf{F}}_p \mathbf{F}^n_p, \ \hat{\mathbf{F}}_p = (\mathbf{I} + \Delta t \frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)) \tag{3.19}$$

3. Transfer the values from particles to the grid and using the boundary conditions to get $\hat{g}_i$,

$$\hat{g}_i = Z_{\mathbf{i}\alpha i}\left(m^n_{\mathbf{i}} v^n_{\mathbf{i}\alpha} + \Delta t f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right)\right) \tag{3.20}$$

where

$$f_{i\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right) = -\sum_p P_{\alpha\beta}(\mathbf{F}_p) F^n_{p\alpha\beta} \frac{\partial N_i}{\partial x_\gamma}(x^n_p) V^0_p \tag{3.21}$$

and $\mathbf{F}_p$ is related to $\mathbf{w}^k$ through Equation (3.35). Note that "P2G" calculates $m_{\mathbf{i}}^n v_{\mathbf{i}\alpha}^n + \Delta t f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right) + \Delta t m_{\mathbf{i}}^n g_\alpha$ and MPM boundary conditions applies $Z_{\mathbf{i}\alpha i}$ in the expression for $\hat{g}_i$ in Equation (3.36).

4. Calculate Newton residual g_i

$$g_i = \frac{Z_{\mathbf{i}\alpha i} m_{\mathbf{i}}^n Z_{\mathbf{i}\alpha j} w_j^k - \hat{g}_i}{\Delta t}. \tag{3.22}$$

Using $\mathbf{w}^k$ defined over the whole grid (i.e. $\mathbf{Z}$ is square). In this case, as long as the initial guess satisfies the boundary conditions, all subsequent Newton iterations $\mathbf{w}^k$ will satisfy the boundary conditions since $\delta\mathbf{w}^k$ will be satisfied by construction. Therefore, there is no need to apply $\mathbf{Z}$ (via MPM boundary conditions) in Equation (3.38) and scale by the mass and divide by Δt.

3.3.7 Newton Differential

$$\delta g_i = \frac{\partial g_i}{\partial \tilde{v}_j}(\mathbf{w}^k)\delta w_j^k$$

1. We compute $\delta\hat{\mathbf{F}}_p$

$$\delta \hat{F}_{p\alpha\beta} = \Delta t Z_{\mathbf{i}\alpha j}\delta w_j^k \frac{\partial N_{\mathbf{i}}}{\partial x_\beta}(\mathbf{x}_p^n) \tag{3.23}$$

$$\delta\mathbf{F}_p = \delta\hat{\mathbf{F}}_p \mathbf{F}_p^n \tag{3.24}$$

2. We then calculate $\delta\hat{f}_{\mathbf{i}\alpha}$ in "P2G" from $\delta\mathbf{F}_p^n$ and $\mathbf{F}_p$ (from Equation (3.35))

$$\delta\hat{f}_i = -\sum_p Z_{\mathbf{i}\alpha i}\frac{\partial P_{\alpha\beta}}{\partial F_{\delta\epsilon}}(\mathbf{F}_p)\delta F_{p\delta\epsilon} F_{p\alpha\beta}^n \frac{\partial N_{\mathbf{i}}}{\partial x_\gamma}(\mathbf{x}_p^n) V_p^0 \tag{3.25}$$

$$\delta g_i = Z_{\mathbf{i}\alpha i}\frac{m_{\mathbf{i}} Z_{\mathbf{i}\alpha j}\delta w_j^k}{\Delta t} - \delta\hat{f}_i \tag{3.26}$$

3.3.8 Explicit Time Discretization

The update of momentum requires that the force is evaluated at t^n for the explicit MPM. The discretize momentum update becomes:

$$\sum_{j=1}^{N_n} M_{ij}(t^n)\frac{\mathbf{v}_j(t^{n+1}) - \mathbf{v}_j(t^n)}{\Delta t} = \mathbf{f}_i(x^n) \tag{3.27}$$

3.3.9 Implicit Discretization

The implicit momentum update requires the forces at the same time as the update. With backward Euler discretization, the conservation of momentum implies that the forces are implicitly dependent on the grid motion.

$$\sum_{j=1}^{N_n} M_{ij}(t^n)\frac{\hat{\mathbf{v}}_j(t^{n+1}) - \mathbf{v}_j(t^n)}{\Delta t} = \mathbf{f}_i(x^{n+1}) = \mathbf{f}_i(x^n + \Delta t \mathbf{v}_j(t^{n+1})) \tag{3.28}$$

with boundary conditions satisfied by

$$\hat{\mathbf{v}}^{n+1} = \mathbf{Z}\tilde{\mathbf{v}}^{n+1}. \tag{3.29}$$

The elastic force $f_{i\alpha}$ is defined as

$$f_{i\alpha}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial \Psi}{\partial F_{\alpha\gamma}}\left(\mathbf{F}_p\left(\hat{\mathbf{x}}\right)\right) F^n_{p\delta\gamma}\frac{\partial N_{\mathbf{i}}}{\partial x_\delta}(\mathbf{x}^n_p)V^n_p \tag{3.30}$$

Solving the systems for $\tilde{\mathbf{v}}^{n+1}$ with the Newton's method.

3.3.10 Residual

For this section $g_i(\tilde{\mathbf{v}})$ is used for residual of the nonlinear equations

$$g_i(\tilde{\mathbf{v}}) = Z_{\mathbf{i}\alpha i}\left(m^n_{\mathbf{i}}\frac{Z_{\mathbf{i}\alpha k}\tilde{v}_k - v^n_{\mathbf{i}\alpha}}{\Delta t} - f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t\mathbf{Z}\tilde{\mathbf{v}}\right) - m^n_{\mathbf{i}}g_\alpha\right) \tag{3.31}$$

where in this case summation on $\mathbf{i}$ and α is implied.

3.3.11 Linearization

The linearization of the residual has the form

$$\frac{\partial g_i}{\partial \tilde{v}_j}(\tilde{\mathbf{v}}) = Z_{\mathbf{i}\alpha i}\left(\frac{m^n_{\mathbf{i}}}{\Delta t}Z_{\mathbf{i}\alpha j} - \Delta t\frac{\partial f_{\mathbf{i}\alpha}}{\partial \hat{x}_{\mathbf{j}\beta}}\left(\mathbf{x} + \Delta t\mathbf{Z}\tilde{\mathbf{v}}\right)Z_{\mathbf{j}\beta j}\right) \tag{3.32}$$

where the linearization of the elastic force satisfies

$$\frac{\partial f_{\mathbf{i}\alpha}}{\partial \hat{x}_{\mathbf{j}\beta}}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial^2 \Psi}{\partial F_{\beta\epsilon} F_{\alpha\gamma}} \left(\mathbf{F}_p\left(\hat{\mathbf{x}}\right)\right) F^n_{p\delta\gamma} F^n_{p\sigma\epsilon} \frac{\partial N_{\mathbf{i}}}{\partial x_\delta}(\mathbf{x}^n_p) \frac{\partial N_{\mathbf{j}}}{\partial x_\sigma}(\mathbf{x}^n_p) V^n_p \tag{3.33}$$

.

3.3.12 Implementation

The formula to update the approximation with Newton's method is

$$\begin{aligned} &\tfrac{\partial g_i}{\partial \tilde{v}_j}(\mathbf{w}^k)\delta w^k_j = -g_i(\mathbf{w}^k) \\ &\mathbf{w}^{k+1} = \mathbf{w}^k + \delta \mathbf{w}^k \end{aligned} \tag{3.34}$$

where $\mathbf{w}^0$ is the initial guess to $\tilde{\mathbf{v}}^{n+1}$ and $\mathbf{w}^k \to \tilde{\mathbf{v}}^{n+1}$.

3.3.13 Newton Residual

The right-hand side of the system for the Newton increment $\delta \mathbf{w}^k$ is computed. We call this computation of the "Newton Residual" $-g_i(\mathbf{w}^k)$:

1. Transfer grid values to particles to compute $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)$ using current guess to grid velocity $\mathbf{w}^k$.

2. Use $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)$ to update $\mathbf{F}_p$:

$$\mathbf{F}_p = \hat{\mathbf{F}}_p \mathbf{F}^n_p, \ \hat{\mathbf{F}}_p = (\mathbf{I} + \Delta t \frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}^n_p)) \tag{3.35}$$

3. Transfer the values from particles to the grid and using the boundary conditions to get $\hat{g}_i$,

$$\hat{g}_i = Z_{\mathbf{i}\alpha i}\left(m^n_{\mathbf{i}} v^n_{\mathbf{i}\alpha} + \Delta t f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right)\right) \tag{3.36}$$

where

$$f_{i\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right) = -\sum_p P_{\alpha\beta}(\mathbf{F}_p) F^n_{p\alpha\beta} \frac{\partial N_i}{\partial x_\gamma}(x^n_p) V^0_p \tag{3.37}$$

and $\mathbf{F}_p$ is related to $\mathbf{w}^k$ through Equation (3.35). Note that "P2G" calculates $m_{\mathbf{i}}^n v_{\mathbf{i}\alpha}^n + \Delta t f_{\mathbf{i}\alpha}\left(\mathbf{x} + \Delta t \mathbf{Z}\mathbf{w}^k\right) + \Delta t m_{\mathbf{i}}^n g_\alpha$ and MPM boundary conditions applies $Z_{\mathbf{i}\alpha i}$ in the expression for $\hat{g}_i$ in Equation (3.36).

4. Calculate Newton residual g_i

$$g_i = \frac{Z_{\mathbf{i}\alpha i} m_{\mathbf{i}}^n Z_{\mathbf{i}\alpha j} w_j^k - \hat{g}_i}{\Delta t}. \tag{3.38}$$

Using $\mathbf{w}^k$ defined over the whole grid (i.e. $\mathbf{Z}$ is square). In this case, as long as the initial guess satisfies the boundary conditions, all subsequent Newton iterations $\mathbf{w}^k$ will satisfy the boundary conditions since $\delta\mathbf{w}^k$ will be satisfied by construction. Therefore, there is no need to apply $\mathbf{Z}$ (via MPM boundary conditions) in Equation (3.38) and scale by the mass and divide by Δt.

3.3.14 Newton Differential

$$\delta g_i = \frac{\partial g_i}{\partial \tilde{v}_j}(\mathbf{w}^k)\delta w_j^k$$

1. We compute $\delta\hat{\mathbf{F}}_p$

$$\delta \hat{F}_{p\alpha\beta} = \Delta t Z_{\mathbf{i}\alpha j}\delta w_j^k \frac{\partial N_{\mathbf{i}}}{\partial x_\beta}(\mathbf{x}_p^n) \tag{3.39}$$

$$\delta\mathbf{F}_p = \delta\hat{\mathbf{F}}_p \mathbf{F}_p^n \tag{3.40}$$

2. We then calculate $\delta\hat{f}_{\mathbf{i}\alpha}$ in "P2G" from $\delta\mathbf{F}_p^n$ and $\mathbf{F}_p$ (from Equation (3.35))

$$\delta\hat{f}_i = -\sum_p Z_{\mathbf{i}\alpha i}\frac{\partial P_{\alpha\beta}}{\partial F_{\delta\epsilon}}(\mathbf{F}_p)\delta F_{p\delta\epsilon} F_{p\alpha\beta}^n \frac{\partial N_{\mathbf{i}}}{\partial x_\gamma}(\mathbf{x}_p^n) V_p^0 \tag{3.41}$$

$$\delta g_i = Z_{\mathbf{i}\alpha i}\frac{m_{\mathbf{i}} Z_{\mathbf{i}\alpha j}\delta w_j^k}{\Delta t} - \delta\hat{f}_i \tag{3.42}$$

CHAPTER 4

A Momentum Conserving Hybrid Particle/Grid Iteration for Volumetric Elastic Contact

Resolving contact and collisions efficiently and accurately is challenging for continuum mechanics simulations. Recently, MPM was used for volumetric elastic contact along finite element meshes [9]. This method resolved collisions efficiently, however, the method did not conserve the global linear and angular momentum. A recent extension to this method that achieves global conservation of linear and angular momentum has been implemented [5]. In addition, this method utilizes augury dynamics to filter the numerical cohesion from MPM that is not in the contact direction. This method is susceptible to a similar limitation to the ones explained in [9]. One of these limitations is that the collisions can be missed when the resolution of the MPM grid is too high. This limitation was overcome by adding resample mass points [9], however this approach violates the basic conservation laws. We propose a resampling scheme that conserves mass, linear and angular momentum that can be used alongside the augury iterations [5] to resolve the collision with the finer resolution of background grids. There is a time step restriction based on the MPM grid that is required to capture the collisions. Using this method alongside impulse-based approaches [10] and implicit time-stepping can capture collisions and conserve linear momentum while having timesteps larger than the restriction imposed by MPM.

We summarize our contributions as:

- An efficient method to simulate volumetric elastic contact.

- A momentum-conserving particle resampling technique for surfaces that conserves the physical representation of the mesh.

4.1 Related Work

Hybrid Eulerian Lagrangian elastic contact: Eulerian techniques for elastic objects with self-collisions were developed by Pai et al. [11]. Li et al. [12] captured the self contact of soft tissues using an Eulerian view. Teng et al. [13] demonstrated that the approach can be coupled naturally with incompressible fluids. Müller et al. [14], Sifakis et al. [15] and Wu et al. [16] mesh the space surrounding elastic objects and enforce positive volume and/or incompressibility constraints on the air surrounding the objects to resolve collisions. Xuchen et al. [9] introduced particles in between the elements to match the Eulerian grid resolution to capture the collision when otherwise would be missed. Tupek et al. [5] proposed an iterative scheme to resolve the contact while eliminating the cohesion while separating.

4.2 Method Overview

A typical approach to making volumetric mesh aware of the collision is to update the boundary velocities. However, this approach will introduce a lagged response to the collision where the material response occurs a timestep behind the collision. This lag in the force can have significant effects on the dynamics of the simulation if the time steps are large. For explicit time discretization schemes that typically require smaller time steps for stability this effect is less severe than implicit. A typical explicit scheme is defined in Fig 4.1 as follows.

- **Normals are updated** The particle normals are updated with the current configuration of the mesh.
- **Lagrangian update on the velocities** The Lagrangian update can be explicit or implicit.

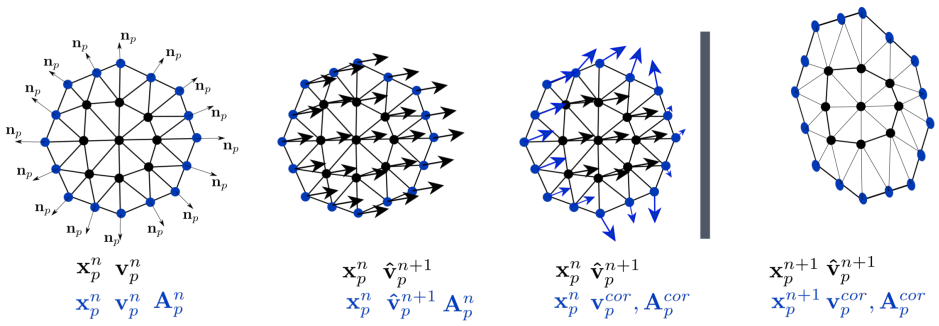


Figure 4.1: **Explicit Collision Method**. The normals are updated, a lagrangian update is made, collision is captured and the positions updated

- **Collision update on velocities** The velocities are updated to account for the collision.
- **Position update on the particles** The positions are updated with the velocities that are aware of collision at the surface, however, the interior nodes are not aware of this collision until the next lagrangian update.

This scheme can work well with explicit time steps, however, the smaller time step required to be stable for stiffer materials is computationally expensive. We leverage the implicit lagrangian time step size by making the lagrangian update aware of the collisions Fig 4.2.

- **Normals are update** The particle normals are updated witht the current configuration of the mesh.
- **Lagrangian update on the velocities** An implicit Lagrangian update on the velocities.
- **Collision update on the velocities** The velocities are updated to account for the collision.

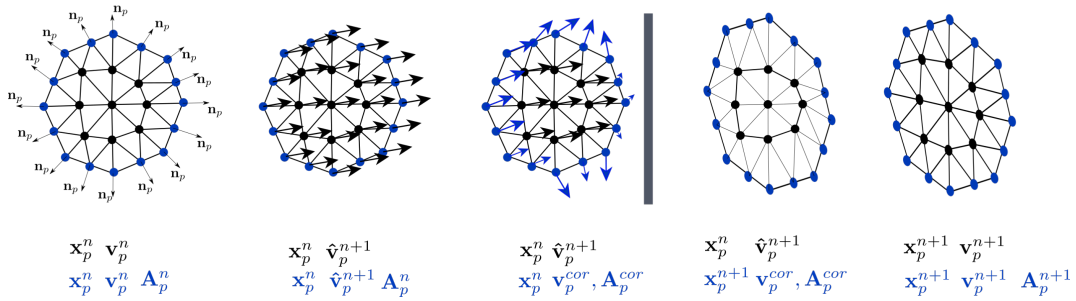


Figure 4.2: **Implicit Collision Method**. The normals are updated, a lagrangian update is made, collision is captured, the boundary is updated and the positions of the surface updated and used a as Dirichlet boundary condition for the lagrangian update of the interior nodes.

- **The surface particles are updated over one timestep.** The surface is updated one timestep.
- **Lagrangian update with the surface as Dirichlet boundary condition** An implicit Lagrangian update on the velocities and the position of the mesh is done with the updated surface as Dirichlet boundary conditions.

4.3 Finite Elements

Our finite element imlementation is similar to [17].The general finite element formulation is derived from the elastic potential energy E due to an arbitrary deformation $\phi(\mathbf{x})$. The energy is the integral of the strain energy density over the continuum domain:

$$E(\phi) = \int_{\Omega} \Psi(\mathbf{F}) d\mathbf{X} \tag{4.1}$$

This energy is discretized in terms of degrees of freedom and the interpolated deformation $\hat{\phi}$.

$$E(\mathbf{x}) = \int_{\Omega} \Psi(\frac{\partial \hat{\phi}}{\partial \mathbf{X}}) d\mathbf{X} \tag{4.2}$$

The discrete energy is used to compute the elastic forces associated to the mesh nodes.

$$\mathbf{f}_i = -\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} \tag{4.3}$$

The domain is discretize with tetahedral elements, each tetahedral has individual contributions to the forces associated to their nodes. The sum of the elements provide the total material response.

$$E(\mathbf{x}) = \sum_e E^e(\mathbf{x}) = \sum_e \int_{\Omega_e} \Psi(\frac{\partial \hat{\phi}}{\partial \mathbf{X}}) d\mathbf{X} \tag{4.4}$$

The nnodal force follows a similar discretization

$$\mathbf{f}_i = \sum_e \mathbf{f}_i^e = -\sum_e \frac{\partial E^e(\mathbf{x})}{\partial \mathbf{x}_i} \tag{4.5}$$

4.4 Linear Tetrahedral Elements

We utilize constant strain tetrahedra based in picewise linear interpolation functions. The deformation gradient can be determined with the local positions $\mathbf{x}$ of the deformed tetrahedral with respect to the underformed positions $\mathbf{X}$ 4.3.

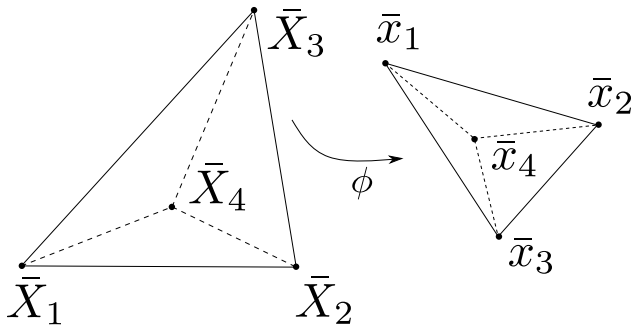


Figure 4.3: **Tetrahedral Element**.The tetrahedral element and the transformation between reference rest state and the current deformed state.

$$\mathbf{F} = \mathbf{D}_s \mathbf{D}_m^{-1} \tag{4.6}$$

$$\mathbf{D}_s = \begin{bmatrix} x_1 - x_4 & x_2 - x_4 & x_3 - x_4 \\ y_1 - y_4 & y_2 - y_4 & y_3 - y_4 \\ z_1 - z_4 & z_2 - z_4 & z_3 - z_4 \end{bmatrix}$$

$$\mathbf{D}_m = \begin{bmatrix} X_1 - X_4 & X_2 - X_4 & X_3 - X_4 \\ Y_1 - Y_4 & Y_2 - Y_4 & Y_3 - Y_4 \\ Z_1 - Z_4 & Z_2 - Z_4 & Z_3 - Z_4 \end{bmatrix}$$

$\mathbf{D}_m$ is only a function of the rest state position. Following [17] the nodal forces can be computed as follows.

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 \end{bmatrix} = -\tfrac{det(\mathbf{D}_m)}{6}\mathbf{P}(\mathbf{F})\mathbf{D}_m^{-T} \\ \mathbf{f}_4 &= -\mathbf{f}_1 - \mathbf{f}_2 - \mathbf{f}_3 \end{aligned} \tag{4.7}$$

Implicit method requires the force differentials $\delta\mathbf{f}$, that required the expression of the stiffness matrix $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$. The force differential is computed directly with information in the current state $\mathbf{x}_*$, the displacement δx.

$$\delta\mathbf{f} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}_*} \cdot \delta\mathbf{x} \tag{4.8}$$

The nodal forces differentials become:

$$\begin{aligned} \delta\mathbf{H} &= \begin{bmatrix} \delta\mathbf{f}_1 & \delta\mathbf{f}_2 & \delta\mathbf{f}_3 \end{bmatrix} = -\tfrac{det(\mathbf{D}_m)}{6}\delta\mathbf{P}(\mathbf{F};\delta\mathbf{F})\mathbf{D}_m^{-T} \\ \delta\mathbf{f}_4 &= -\delta\mathbf{f}_1 - \delta\mathbf{f}_2 - \delta\mathbf{f}_3 \end{aligned} \tag{4.9}$$

The differential deformation gradient $\delta\mathbf{F}$ is:

$$\delta\mathbf{F} = \delta\mathbf{D}_s\mathbf{D}_m^{-1} \tag{4.10}$$

$$\delta\mathbf{D}_s = \begin{bmatrix} \delta x_1 - \delta x_4 & \delta x_2 - \delta x_4 & \delta x_3 - \delta x_4 \\ \delta y_1 - \delta y_4 & \delta y_2 - \delta y_4 & \delta y_3 - \delta y_4 \\ \delta z_1 - \delta z_4 & \delta z_2 - \delta z_4 & \delta z_3 - \delta z_4 \end{bmatrix}$$

The Piola Kirchhoff $\mathbf{P}(\mathbf{F})$ is derived from the constitutive model, this allow us to model different materials. The differential of the Green strain E tensor is used:

$$\begin{aligned}\mathbf{E} &= \tfrac{1}{2}(\mathbf{F}^T\mathbf{F} - \mathbf{I}) \\ \delta\mathbf{E} &= \tfrac{1}{2}(\delta\mathbf{F}^T\mathbf{F} + \mathbf{F}^T\delta\mathbf{F})\end{aligned} \tag{4.11}$$

Example for St. Venant-Kirchhoff materials.

$$\begin{aligned}\mathbf{P}(\mathbf{F}) &= \mathbf{F}[2\mu\mathbf{E} + \lambda tr(\mathbf{E})\mathbf{I}] \\ \delta\mathbf{P}(\mathbf{F};\delta\mathbf{F}) &= \delta\mathbf{F}[2\mu\mathbf{E} + \lambda tr(\mathbf{E})\mathbf{I}] + \mathbf{F}[2\mu\delta\mathbf{E} + \lambda tr(\delta\mathbf{E})\mathbf{I}]\end{aligned} \tag{4.12}$$

4.4.0.1 Backward Euler

Backward Euler requires that the positions and velocities at the current time providing a set of nonlinear equations.

$$\begin{aligned}\mathbf{x}^{n+1} &= \mathbf{x}^n + \Delta t\mathbf{v}^{n+1} \\ \mathbf{v}^{n+1} &= \mathbf{v}^n + \Delta t\mathbf{M}^{-1}(\mathbf{f}_{elastic}(\mathbf{x}^{n+1}) + \mathbf{f}_{damping}(\mathbf{x}^{n+1}, \mathbf{v}^{n+1}))\end{aligned} \tag{4.13}$$

The equations are linearize around iterations that will correct the solution.

$$\begin{aligned}\Delta\mathbf{x} &= \mathbf{x}^{n+1}_{(k+1)} - \mathbf{x}^{n+1}_{(k)} \\ \Delta\mathbf{v} &= \mathbf{v}^{n+1}_{(k+1)} - \mathbf{v}^{n+1}_{(k)} \\ \Delta\mathbf{x} &= \Delta t\Delta\mathbf{v}\end{aligned} \tag{4.14}$$

Proportional Rayleigh damping in the stiffness matrix can be incomporated. Since the damping force $\mathbf{f}_{damping}$ will depend on the current velocity. The damping coefficient γ can be used to model structural damping having typical values lower than 0.1.

$$\mathbf{v}^{n+1}_{(k)} + \Delta\mathbf{v} = \mathbf{v}^n + \Delta t\mathbf{M}^{-1}(\mathbf{f}_e(\mathbf{x}^{n+1}) + \tfrac{\partial\mathbf{f}_e}{\partial\mathbf{x}}|_{\mathbf{x}^{n+1}_{(k)}} \cdot \Delta\mathbf{x} - \gamma(-\tfrac{\partial\mathbf{f}_e}{\partial\mathbf{x}}|_{\mathbf{x}^{n+1}_{(k)}})(\mathbf{v}^{n+1}_{(k)} + \Delta\mathbf{v}) \tag{4.15}$$

For notation we use the stiffness matrix $\mathbf{K}$ istead of $-\frac{\partial\mathbf{f}_e}{\partial\mathbf{x}}$. Further manipulation results in the following symetric positive deffinite system that can be solve via Conjugate gradients.

$$
\begin{aligned}
&[(1+\tfrac{\gamma}{\Delta t})\mathbf{K}(\mathbf{x}_{(k)}^{n+1})+\tfrac{1}{\Delta t^2}\mathbf{M}]\Delta\mathbf{x}=\tfrac{1}{\Delta t}\mathbf{M}(\mathbf{v}^n-\mathbf{v}_{(k)}^{n+1})+\mathbf{f}_e(\mathbf{x}_{(k)}^{n+1})-\gamma\mathbf{K}(\mathbf{x}_{(k)}^{n+1}\mathbf{v}_{(k)}^{n+1})\\
&\mathbf{v}_{(k+1)}^{n+1}=\mathbf{v}_{(k)}^{n+1}+\tfrac{1}{\Delta t}\Delta\mathbf{x}
\end{aligned}
\tag{4.16}
$$

4.4.1 Lagrangian Update

To resolve the collision we apply two Lagrangian to the material as described by 4.16. The first one is used to update the velocities $\mathbf{v}^{n+1}$, but the particles are not advanced during this velocity update. The velocities at the boundary are made aware of the collision $\hat{\mathbf{v}}^{n+1}$, with the boundary particles being updated $\mathbf{x}^{n+1}$ and used as Dirichlet boundary conditions to solve for the interior positions $\mathbf{x}^{n+1}$ and velocities $\mathbf{v}^{n+1}$.

4.4.2 Collision Update

The collision update on the velocities is done through an MPM grid this requires that the surface particles store APIC affine to conserve angular momentum. Resampling is done to avoid missing the collision on finer MPM grid resolutions.

4.4.2.1 Resampling

Increasing the grid resolution will decrease the distance where the transfer of momentum is done and will better resolve the contact. In Fig 4.4 we can show how the method can miss the collision with no resampling as the resolution of the background grid is increased. Resampling is done to avoid missing the collision Fig.4.5.

Resample points positions are generated accordingly to capture the collision on the background grid. The set of resampling points include the original points in the element. A portion of the mass is taken from the original points and given to the resample points. Conservation of the center of mass of the resample points is similar to having the resampling points integrate a linear function exactly while all the particles have positive weights. This

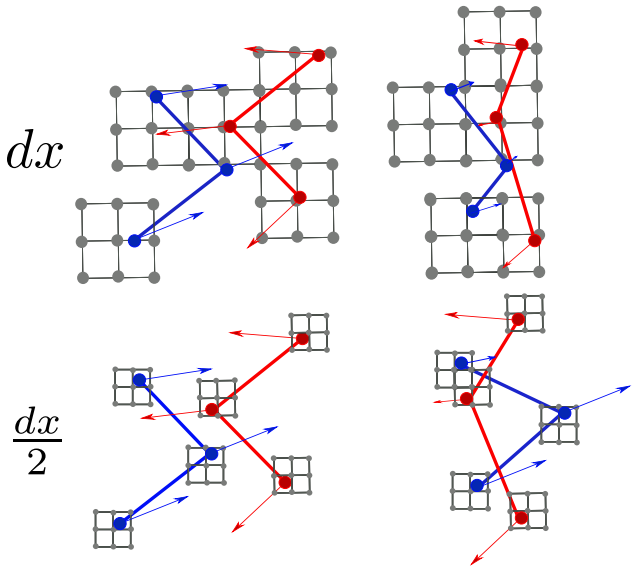


Figure 4.4: **Collision Missed**. **Top** The collision is captured with an appropriate grid resolution. **Bot** Under refinement the collision can be missed.

requirement can be satisfied with local maximum-entropy shape functions [18] that can be used to determine the appropriate mass given to the resample points.

4.4.2.2 Local Maximum-Entropy Approximation

The max-entropy shape functions are calculated via a maximization problem done on the measure of uncertainty or information entropy. Bias on the function is introduced with [18] in terms of β when β equals zero the algorithm provides the least bias weight functions. The multiple solutions to the system are all plausible, however not all particle positions will provide a solution. Poor particle distribution can cause some of the weights to be zero or if the distribution is not possible to sustain a solution the system will not reach a solution.

$$H(\mathbf{p}) = -\sum_{a=1}^{N} p_a \log p_a + \beta \sum_{a=1}^{N} p_a |\mathbf{x} - \mathbf{x}_a|^2 \tag{4.17}$$

This function subjected to the following constraints:

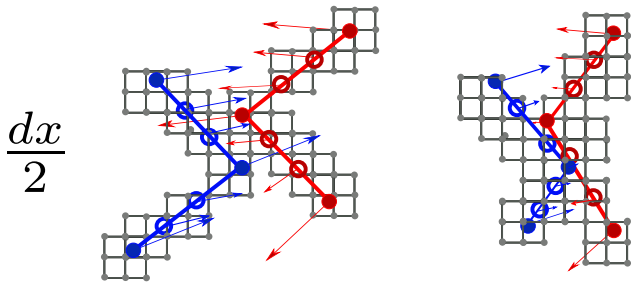


Figure 4.5: **Collision Capture**. The collision is captured under refinement via the resampling of particles.

The function weights p_a must be greater than zero

$$p_a => 0, a = 1, ..., N \tag{4.18}$$

The function weights must satisfy partition of unity and reproduction of linear functions

$$\sum_{a=1}^{N} p_a = 1 \tag{4.19}$$

$$\sum_{a=1}^{N} p_a \mathbf{x}_a = \mathbf{x} \tag{4.20}$$

The Langrangian associated to the least based weights uses lagrangian multipliers λ_0 to enforce and constraints becomes:

$$L(\mathbf{p}, \lambda_0, \boldsymbol{\lambda}) = H(\mathbf{p}) + \lambda_0(1 \cdot \mathbf{p} - 1) + \boldsymbol{\lambda} \cdot \sum_a p_a(\mathbf{x}_a - \mathbf{x}) \tag{4.21}$$

[18] defines a partition function Z

$$Z(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{a=1}^{N} \exp[-\beta|\mathbf{x} - \mathbf{x}_a|^2 + \boldsymbol{\lambda} \cdot (\mathbf{x} - \mathbf{x}_a)] \tag{4.22}$$

The unique solution to the local max-ent problem shape function values p_a, given an specific β becomes $p_{\beta\alpha}$. Here we note that $\beta = 0$ will tend to spread out the weights and that larger values of β will localize the distribution of the weights. Nevertheless, all of the unique

solutions of the system that depend on β will satisfy partition of unity, weights greater than zero, and reproduce linear functions.

$$p_{\beta\alpha} = \frac{1}{Z(\mathbf{x}, \boldsymbol{\lambda}^*)} \exp[-\beta|\mathbf{x} - \mathbf{x}_a|^2 + \boldsymbol{\lambda} \cdot (\mathbf{x} - \mathbf{x}_a)] \tag{4.23}$$

Where,

$$\boldsymbol{\lambda}^* = argmin \log Z(\mathbf{x}, \boldsymbol{\lambda}) \tag{4.24}$$

A unique solution for a value of β can be found by solving for the nonlinear equation 4.25 using Newton-Rapshon iterations where $\mathbf{r}$ is the gradient of the objective function and $\mathbf{J}$ is the Hessian of the objective function.

$$\mathbf{r}(\mathbf{x}, \boldsymbol{\lambda}) = 0 \tag{4.25}$$

$$\mathbf{r}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{a=1}^{N} p_a(\mathbf{x} - \mathbf{x}_a) \tag{4.26}$$

$$\mathbf{J}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{a=1}^{N} p_a(\mathbf{x} - \mathbf{x}_a) \otimes (\mathbf{x} - \mathbf{x}_a) - \mathbf{r}(\mathbf{x}, \boldsymbol{\lambda}) \otimes \mathbf{r}(\mathbf{x}, \boldsymbol{\lambda}) \tag{4.27}$$

The weights will depend on the center of mass of the original particles and the resample particle distribution. The original particles distribute a percentage of their mass to the center of mass location. This mass is distributed to the resample particles that get the weights with the max-entropy approximation evaluated at the center of mass. This can be done as in Fig.4.6 in 2D for the surface segments and in 3D with the surface faces, the particle position can be determined with Poisson disk sampling. This method can also handle structured particle positions as seen in Fig.4.7

Here we note that conservation of the inertia tensor principal axes of rotation requires the weights of the particles to interpolate exactly quadratic functions. For arbitrary locations, the constraint of the system that guarantees the second-order property is not possible

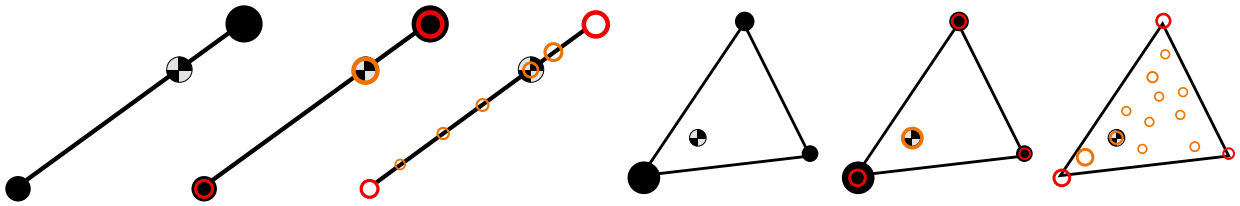


Figure 4.6: **Max Entropy Resampling**. (**Left**) A segment distribute a portion of their masses to the center of mass and then distributes them to the resample particles. (**Right**) A surface triangle distributes a portion of their masses to the center of mass and then distributes them to the resample particles.

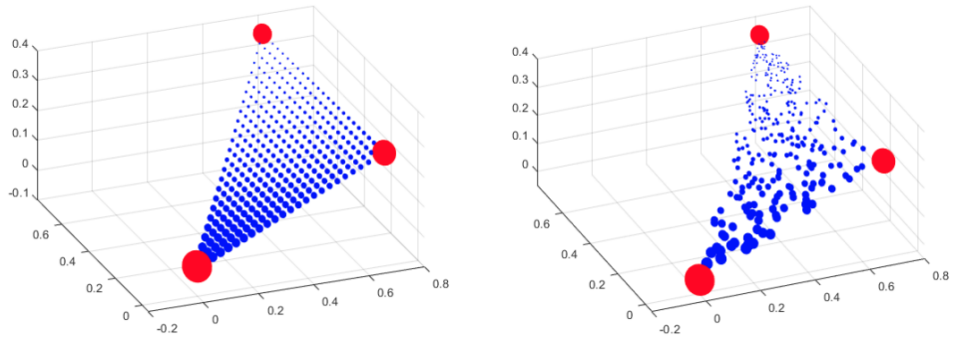


Figure 4.7: **Max Entropy Particle Sampling**. We show the weights of the resample particles and the support of arbitrary position placement.

to achieve. A system that approximately reproduces second-order functions through an additional constraint has been derived by Aroyo and Ortiz [18]. This is done through a gap function $g(\mathbf{x})$ used in the constraint instead of the exact quadratic function.

The system in 4.17 includes the following extra constraint:

$$\sum_{a=1}^{N} p_a(\mathbf{x} - \mathbf{x}_a)^2 = g(\mathbf{x}) \tag{4.28}$$

The partition function becomes, where $\boldsymbol{\mu}$ is the Lagrange multiplier for the quadratic constraint:

$$Z(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{a=1}^{N} \exp[-\beta|\mathbf{x} - \mathbf{x}_a|^2 + \boldsymbol{\lambda} \cdot (\mathbf{x} - \mathbf{x}_a) + \boldsymbol{\mu} : [(\mathbf{x} - \mathbf{x}_a) \otimes (\mathbf{x} - \mathbf{x}_a) - \mathbf{g}(\mathbf{x})]] \quad (4.29)$$

And the unique solution becomes

$$p_{\beta\alpha} = \frac{1}{Z(\mathbf{x}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)} \exp[-\beta|\mathbf{x} - \mathbf{x}_a|^2 + \boldsymbol{\lambda} \cdot (\mathbf{x} - \mathbf{x}_a) + \boldsymbol{\mu}^* : [(\mathbf{x} - \mathbf{x}_a) \otimes (\mathbf{x} - \mathbf{x}_a) - \mathbf{g}(\mathbf{x})]] \quad (4.30)$$

4.4.2.3 Boundary Element Mass and Momentum

Each element position $\mathbf{x}^e_{i^e}$ with mass $m^e_{i^e}$ and APIC state $\mathbf{v}^e_{i^e}$ and $\mathbf{A}^e_{i^e}$ has associated grid momenta $\mathbf{p}^{e,i^e}_{\mathbf{i}} = m^e_{i^e}\left(\mathbf{v}^e_{i^e} + \mathbf{A}^e_{i^e}\left(\mathbf{x}_{\mathbf{i}} - \mathbf{x}^e_{i^e}\right)\right) N_{\mathbf{i}}(\mathbf{x}^e_{i^e})$ at grid nodes $\mathbf{x}_{\mathbf{i}}$ defined from the APIC transfer to the grid. The sum of these defines the boundary element grid momentum distribution $\mathbf{p}^e_{\mathbf{i}} = \sum_{i^e=0}^{d-1} \mathbf{p}^{e,i^e}_{\mathbf{i}}$. The total linear $\mathbf{p}^e = \sum_{\mathbf{i}} \mathbf{p}^e_{\mathbf{i}}$ and angular $\mathbf{l}^e = \sum_{\mathbf{i}} (\mathbf{x}_{\mathbf{i}} - \mathbf{x}^e_{\text{com}}) \times \mathbf{p}^e_{\mathbf{i}}$ (computed about element center of mass $\mathbf{x}^e_{\text{com}} = \frac{1}{m^e} \sum_{i^e=0}^{d-1} m^e_{i^e} \mathbf{x}^e_{i^e}$ where $m^e = \sum_{i^e} m^e_{i^e}$ is the total element mass) momenta of the element are defined from the element grid momentum distribution. Furthermore, the total linear $\mathbf{p} = \sum_e \mathbf{p}^e$ and angular momentum $\mathbf{l} = \sum_e \mathbf{l}^e + (\mathbf{x}^e_{\text{com}} - \mathbf{x}_{\text{com}}) \times \mathbf{p}^e$ of the boundary is defined from the element-wise momenta $\mathbf{p}^e$ and $\mathbf{l}^e$ (where $\mathbf{x}_{\text{com}}$ is the center of mass of the boundary mesh). We design our resampling strategy to produce new state $\hat{\mathbf{x}}^e_{j^e}$, $\hat{m}^e_{j^e}$, $\hat{\mathbf{v}}^e_{j^e}$, $\hat{\mathbf{A}}^e_{j^e}$ for $0 \leq j^e < N^e_r$ that preserves the element-wise momenta $\mathbf{p}^e$ and $\mathbf{l}^e$.

4.4.2.4 Mass Resampling: Partition of Unity and Linear Reproduction

We choose the resampled positions $\hat{\mathbf{x}}^e_{j^e}$ and masses $\hat{m}^e_{j^e}$ in a manner that guarantees linear reproduction and partition of unity properties

$$\sum_{j^e=0}^{N^e_r-1} \hat{m}^e_{j^e} \lambda^e_{j^e i^e} = m^e_{i^e} \tag{4.31}$$

$$\sum_{j^e=0}^{N^e_r-1} \hat{m}^e_{j^e} = \sum_{i^e=0}^{d-1} m^e_{i^e} \tag{4.32}$$

where $\lambda^e_{j^e i^e}$ are the barycentric weights of the resampled positions $\hat{\mathbf{x}}^e_{j^e}$ relative to $\mathbf{x}^e_{i^e}$. When possible , this can be guaranteed by choosing $\hat{m}^e_{j^e} = \sum_{k^e=0}^{d-1} m^e_{k^e} \hat{N}^e_{j^e}(\mathbf{x}^e_{k^e})$ with interpolating functions $\hat{N}^e_{j^e}$ associated with resample position $\hat{\mathbf{x}}^e_{j^e}$ that satisfy analogous properties. Equation (4.31) is satisfied from

$$\sum_{j^e=0}^{N^e_r-1} \hat{m}^e_{j^e} \lambda^e_{j^e i^e} = \sum_{j^e=0}^{N^e_r-1} \sum_{k^e=0}^{d-1} m^e_{k^e} \hat{N}^e_{j^e}(\mathbf{x}^e_{k^e}) \lambda^e_{j^e i^e} \tag{4.33}$$

$$= \sum_{k^e=0}^{d-1} m^e_{k^e} \sum_{j^e=0}^{N^e_r-1} \lambda^e_{j^e i^e} \hat{N}^e_{j^e}(\mathbf{x}^e_{k^e}) \tag{4.34}$$

$$= \sum_{k^e=0}^{d-1} m^e_{k^e} \lambda^e_{k^e i^e} \tag{4.35}$$

$$= m^e_{i^e} \tag{4.36}$$

assuming the linear reproduction property $\sum_{j^e=0}^{N^e_r-1} \lambda^e_{j^e i^e} \hat{N}^e_{j^e}(\mathbf{x}^e_{k^e}) = \lambda^e_{k^e i^e}$ (since the barycentric weights are linear in positions $\hat{\mathbf{x}}^e_{j^e}$) and since $\lambda^e_{k^e i^e} = \delta_{k^e i^e}$ because the barycentric weights of element positions with respect to themselves are either zero or 1. Furthermore Equa-

tion (4.31) is satisfied

$$\sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e = \sum_{j^e=0}^{N_r^e-1} \sum_{k^e=0}^{d-1} m_{k^e}^e \hat{N}_{j^e}^e(\mathbf{x}_{k^e}^e) \tag{4.37}$$

$$= \sum_{k^e=0}^{d-1} m_{k^e}^e \sum_{j^e=0}^{N_r^e-1} \hat{N}_{j^e}^e(\mathbf{x}_{k^e}^e) \tag{4.38}$$

$$= \sum_{k^e=0}^{d-1} m_{k^e}^e \tag{4.39}$$

assuming the partition of unity property $\sum_{j^e=0}^{N_r^e-1} \hat{N}_{j^e}^e(\mathbf{x}) = 1$. We note that Equation (4.32) guarantees that resampling preserves the total mass and Equation (4.31) guarantees preservation of the center of mass since $\sum_{i^e=0}^{d-1} \lambda_{j^e i^e}^e \mathbf{x}_{i^e}^e = \hat{\mathbf{x}}_{j^e}^e$.

Arroyo and Ortiz [18] provide a general method for designing interpolation functions $\hat{N}_{j^e}^e$ associated with unstructured resample positions $\hat{\mathbf{x}}_{j^e}^e$ that satisfy the appropriate partition of unity and linear reproduction properties. However, we found that a simple strategy suffices for our purposes. We choose the first d resample points to be the original points of the boundary element $\hat{\mathbf{x}}_{k^e}^e = \mathbf{x}_{k^e}^e$, $0 \leq k^e < d$. We choose the next resample point to be the center of mass of the element $\hat{\mathbf{x}}_d^e = \mathbf{x}_{\text{com}}^e$. We set $\hat{m}_{k^e}^e = \frac{m_{k^e}^e}{2}$, $0 \leq k^e < d$ which preserves their center of mass as $\mathbf{x}_{\text{com}}^e$ and assigns them half the total element mass $\sum_{k^e=0}^{d-1} \hat{m}_{k^e}^e = \frac{m^e}{2}$. The additional resample points $\hat{\mathbf{x}}_{j^e}^e$, $j^e > d$ are added and assigned masses $\hat{m}_{j^e}^e$, $j^e > d$ such that their center of mass is equal to the element center of mass and their total mass is equal to one-quarter of the total element mass

$$\frac{1}{\sum_{j^e=d+1}^{N_r^e-1} \hat{m}_{j^e}^e} \sum_{j^e=d+1}^{N_r^e-1} \hat{m}_{j^e}^e \hat{\mathbf{x}}_{j^e}^e = \mathbf{x}_{\text{com}}^e \tag{4.40}$$

$$\sum_{j^e=d+1}^{N_r^e-1} \hat{m}_{j^e}^e = \frac{m^e}{4}. \tag{4.41}$$

Lastly, the center of mass resample point $\hat{\mathbf{x}}_d^e$ is assigned the remaining quarter of the element mass $\hat{m}_d^e = \frac{m^e}{4}$. This resampling strategy satisfies the conditions in Equations (4.31) and Equation (4.32). We note that although the strategy of Arroyo and Ortiz [18] suffices for

satisfying these conditions, it does not guarantee that the center of mass is preserved by the resample points $d+1 \leq j^e < N_r^e$ (Equation (4.40)). This property is essential for condensing APIC state back into the original boundary state at the end of grid transfers.

4.4.2.5 Momentum Resampling

We choose the resampled APIC velocity state $\hat{\mathbf{v}}_{j^e}^e$ and $\hat{\mathbf{A}}_{j^e}^e$ in a manner that preserves the total linear and angular momenta of the boundary element and that preserves the state of the first d resample points $\hat{\mathbf{x}}_{k^e}^e$, $0 \leq k^e < d$ that coincide with the original boundary positions $\mathbf{x}_{i^e}^e$, $0 \leq i^e < d$. Recall that the linear and angular momentum of the original points in the element are defined from the grid momentum distributions $\mathbf{p}_{\mathbf{i}}^{e,i^e} = m_{i^e}^e \left(\mathbf{v}_{i^e}^e + \mathbf{A}_{i^e}^e \left(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_{i^e}^e\right)\right) N_{\mathbf{i}}(\mathbf{x}_{i^e}^e)$ and that the total linear and angular momenta of the element are

$$\mathbf{p}^e = \sum_{i^e=0}^{d-1} m_{i^e}^e \mathbf{v}_{i^e}^e \tag{4.42}$$

$$\mathbf{l}^e = \sum_{i^e=0}^{d-1} \frac{\Delta x^2 m_{i^e}^e}{4} \boldsymbol{\epsilon} : \mathbf{A}_{i^e}^{e\,T} + \left(\mathbf{x}_{i^e}^e - \mathbf{x}_{\text{com}}^e\right) \times m_{i^e}^e \mathbf{v}_{i^e}^e \tag{4.43}$$

We choose our resampled APIC state to preserve these quantities $\hat{\mathbf{p}}^e = \mathbf{p}^e$ and $\hat{\mathbf{l}}^e = \mathbf{l}^e$

$$\hat{\mathbf{p}}^e = \sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e \hat{\mathbf{v}}_{j^e}^e \tag{4.44}$$

$$\hat{\mathbf{l}}^e = \sum_{j^e=0}^{N_r^e-1} \frac{\Delta x^2 \hat{m}_{j^e}^e}{4} \boldsymbol{\epsilon} : \hat{\mathbf{A}}_{j^e}^{e\,T} + \left(\hat{\mathbf{x}}_{j^e}^e - \mathbf{x}_{\text{com}}^e\right) \times \hat{m}_{j^e}^e \hat{\mathbf{v}}_{j^e}^e. \tag{4.45}$$

We resample linear velocities $\hat{\mathbf{v}}_{j^e}^e$ using linear interpolation

$$\hat{\mathbf{v}}_{j^e}^e = \sum_{i^e=0}^{d-1} \lambda_{j^e i^e}^e \mathbf{v}_{i^e}^e. \tag{4.46}$$

This preserves the total linear momentum by the linear reproduction property in Equation (4.31)

$$\sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e \hat{\mathbf{v}}_{j^e}^e = \sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e \sum_{i^e=0}^{d-1} \lambda_{j^e i^e}^e \mathbf{v}_{i^e}^e \tag{4.47}$$

$$= \sum_{i^e=0}^{d-1} \mathbf{v}_{i^e}^e \sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e \lambda_{j^e i^e}^e \tag{4.48}$$

$$= \sum_{i^e=0}^{d-1} m_{i^e}^e \mathbf{v}_{i^e}^e \tag{4.49}$$

$$= \mathbf{p}^e. \tag{4.50}$$

Furthermore, the linear velocities of the original element points are preserved $\hat{\mathbf{v}}_{k^e}^e = \mathbf{v}_{k^e}^e$, $0 \leq k^e < d$. We also define resampled affine velocities using linear interpolation, but with a corrective angular velocity $\Delta\boldsymbol{\omega}^e$ for resample points $d \leq j^e < N_r^e$ chosen to allow for perfect conservation of angular momentum

$$\hat{\mathbf{A}}_{k^e}^e = \mathbf{A}_{k^e}^e,\ 0 \leq k^e < d \tag{4.51}$$

$$\hat{\mathbf{A}}_{j^e}^e = \sum_{i^e=0}^{d-1} \lambda_{j^e i^e}^e \mathbf{A}_{i^e}^e + \boldsymbol{\epsilon}^T : \Delta\boldsymbol{\omega}^e,\ d \leq j^e < N_r^e. \tag{4.52}$$

Here the notation $\boldsymbol{\epsilon}^T : \Delta\boldsymbol{\omega}^e$ refers to the second order tensor $\mathbf{A}$ with indices $A_{\alpha\beta} = \epsilon_{\gamma\alpha\beta}\omega_\gamma^e$. The corrective angular velocity $\Delta\boldsymbol{\omega}^e$ affects the total angular momentum of the resampled element as

$$\sum_{j^e=0}^{N_r^e-1} \hat{m}_{j^e}^e \frac{\Delta x^2}{4} \boldsymbol{\epsilon} : \hat{\mathbf{A}}_{j^e}^{e\,T} + \left(\hat{\mathbf{x}}_{j^e}^e - \mathbf{x}_{\text{com}}^e\right) \times \hat{m}_{j^e}^e \hat{\mathbf{v}}_{j^e}^e = \tag{4.53}$$

$$\sum_{k^e=0}^{d-1} \hat{m}_{k^e}^e \frac{\Delta x^2}{4} \boldsymbol{\epsilon} : \mathbf{A}_{k^e}^{e\,T} + \left(\mathbf{x}_{k^e}^e - \mathbf{x}_{\text{com}}^e\right) \times \hat{m}_{k^e}^e \mathbf{v}_{k^e}^e + \tag{4.54}$$

$$\sum_{j^e=d}^{N_r^e-1} \sum_{i^e=0}^{d-1} \frac{\Delta x^2}{4} \hat{m}_{j^e}^e \lambda_{j^e i^e}^e \boldsymbol{\epsilon} : \mathbf{A}_{i^e}^{e\,T} + \left(\hat{\mathbf{x}}_{j^e}^e - \mathbf{x}_{\text{com}}^e\right) \times \hat{m}_{j^e}^e \lambda_{j^e i^e}^e \mathbf{v}_{i^e}^e + \tag{4.55}$$

$$\sum_{j^e=d}^{N_r^e-1} \hat{m}_{j^e}^e \frac{\Delta x^2}{4} \Delta\boldsymbol{\omega}^e. \tag{4.56}$$

Therefore, we choose the corrective angular velocity as

$$\Delta\boldsymbol{\omega}^e = \frac{4}{\Delta x^2 \sum_{j^e=d}^{N_r^e-1} \hat{m}^e_{j^e}} \Delta \mathbf{l}^e \tag{4.57}$$

$$\Delta \mathbf{l}^e = \mathbf{l}^e - \sum_{k^e=0}^{d-1} \hat{m}^e_{k^e} \frac{\Delta x^2}{4} \boldsymbol{\epsilon} : \mathbf{A}^{e\,T}_{k^e} + (\mathbf{x}^e_{k^e} - \mathbf{x}^e_{\text{com}}) \times \hat{m}^e_{k^e} \mathbf{v}^e_{k^e} \tag{4.58}$$

$$- \sum_{j^e=d}^{N_r^e-1} \sum_{i^e=0}^{d-1} \frac{\Delta x^2}{4} \hat{m}^e_{j^e} \lambda^e_{j^e i^e} \boldsymbol{\epsilon} : \mathbf{A}^{e\,T}_{i^e} + \left(\hat{\mathbf{x}}^e_{j^e} - \mathbf{x}^e_{\text{com}}\right) \times \hat{m}^e_{j^e} \lambda^e_{j^e i^e} \mathbf{v}^e_{i^e}. \tag{4.59}$$

Note that the affine velocities $\hat{\mathbf{A}}^e_{j^e}$ do not affect to total linear momentum of the element and that therefore, our resampling strategy perfectly conserves the total linear and angular momentum of the element.

4.4.3 P2G

We introduce APIC affine information that will be updated through the simulation. The resampling is done to conserve mass and momentum and the particle to grid operation is done in the standard APIC way.

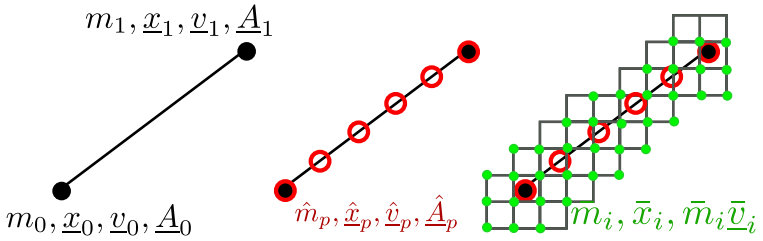


Figure 4.8: **P2G with Resampling**. **Left** The original particles store APIC information. **Middle** The particles are resampled conserving the original particles linear and affine information.**Right** The resample particles information is transfered to the background grid using APIC transfers.

4.4.4 Merging

The total linear and angular momentum of the resample particles around the surface element center of mass is calculated. The MPM grid inertia tensor representation of the original element masses is computed. This is used to transfer the angular momentum of the resample particles to the original splatted particles. This operation does not generate any linear momentum changes. Similarly, the mass-weighted linear momentum of the resample particles is transferred to the original particle MPM grid. This operation does not generate any changes in angular momentum. The merge process is shown in Fig.4.9.

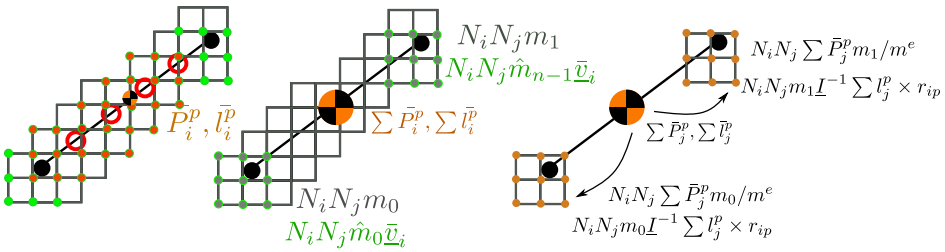


Figure 4.9: **Grid to Particle Merge**.**Left** The linear and angular momentum around the center of mass of the element of the particles is obtained through the APIC transfers. **Middle** The resampled particles that are not in the set of original nodes add their total linear and angular momentum to the center of mass. **Right** Their linear and angular momentum is transferred from the center of mass to the original particles' background grid. This is done with the inverse inertia tensor of the background grid of the original points and a mass ratio for the momentum.

4.4.5 G2P

Once the resample particles background grid is merged into the original particle background grid. The grid to particle operation is done in the standard APIC way. This operation uses the original masses and the original masses splatted weights to update the particle state 4.11.

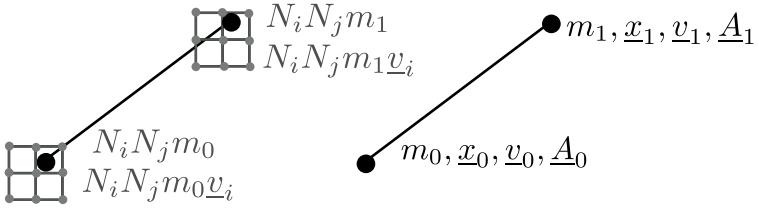


Figure 4.10: **Grid to Particle**.The merged background grid of the original particles transfer the information to the original points with standard APIC transfers.

4.4.6 Augury Interaions

1. Compute $\hat{\mathbf{v}}_p = \mathbf{v}_p^n + \frac{\Delta t}{m_p}\mathbf{f}_p(\mathbf{x}^{n+k})$ ($k = 0$: symplectic Euler, $k = 1$: backward Euler).

2. Using $\hat{\mathbf{v}}$ to denote the vector of all updated boundary particles velocities and $\mathbf{M}$ for the process of going from P2G followed by G2P, define the initial guess for the augury corrected boundary particle velocities as $\mathbf{v}^{\text{cor}} = \mathbf{M}\hat{\mathbf{v}}$.

3. Iterative Augury ($k = 0, 1, \ldots$)

   (a) $\delta\mathbf{v} = \mathbf{v}^{\text{cor}} - \hat{\mathbf{v}}$.

   (b) If trial delta is cohesive ($\delta\mathbf{v}_{p^b} \cdot \mathbf{n}_{p^b} > 0$), $\delta\mathbf{v}_{p^b} = \delta\mathbf{v}_{p^b}$, otherwise $\delta\mathbf{v}_{p^b} = \mathbf{0}$.

   (c) $\mathbf{v}^{\text{cor}} += (\mathbf{M} - \mathbf{I})\,\delta\mathbf{v}$.

4. $\mathbf{v}_{p^b}^{n+1} = \mathbf{v}_{p^b}^{\text{cor}}$.

5. Update positions for primary particles: $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t\mathbf{v}_p^{n+1}$.

Remarks:

1. Note that the operation $\mathbf{Mq}$ conserves the linear and angular momentum of the boundary particle state.

4.4.6.1 Numerical Cohesion

MPM numerical cohesion occurs when the grid velocity of the particles is merged and the exchange in momentum blends the information. This type of cohesion is part of the mechanism that resolves the collision between particles. When the grid resolution has increased the influence of the cohesion is decreases, but it remains present. In Fig. 4.11 the interpolated velocity that comes from MPM transfer is shown in black. This interpolation will prevent separation, when utilizing augury and a small grid the cohesion can be fully removed for traditional MPM. However, cohesion can still affect MPM interior particles if they do not possess a normal to filter the collision. For volumetric meshes collision, the boundary is only simulated having this type of iterations handle the separation appropriately.

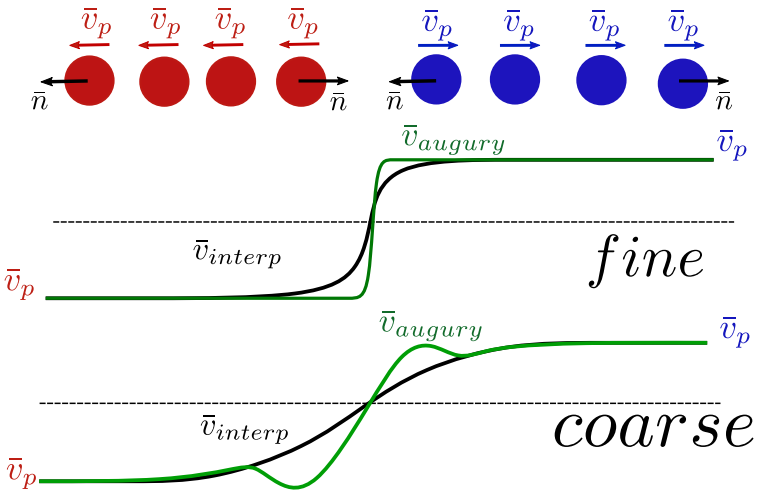


Figure 4.11: **Cohesion** Augury is used in the surface particles were traditional MPM interpolated velocities (black) whercan be seen in the coarse grid from the collision of separating objects.

4.4.6.2 Convergence

Multiple iterations are required to filter the cohesion of the background grid. The number of iterations required depends on the background grid coupling and how many grid nodes

interact with each other. In Fig.4.12 typical convergence of two separating particles is presented in 3D. The spacing of the particles is the same and the grid resolution is varied. Were the particles fully interact with each other convergence is fast required around 5 augury iterations. coarse dx corresponds to a case where the particles overlap and interact with 2/3 of the MPM background grid. Convergence requires around 10 augury iterations and the magnitude of the changes are relatively large. The fine dx has the particles overlap with 1/3 of the MPM background grid. As a result, the needed change is smaller, but due to the interaction being less coupled convergence is much slower than the coarser case. In this case, convergence required around 50 augury iterations. Here we note that each iteration cost is similar to an MPM P2G-G2P operation around all the boundary particles. On larger systems requiring multiple augury iterations can hinder the performance. Nevertheless, in practice 5 iterations can show a significant reduction in cohesion.

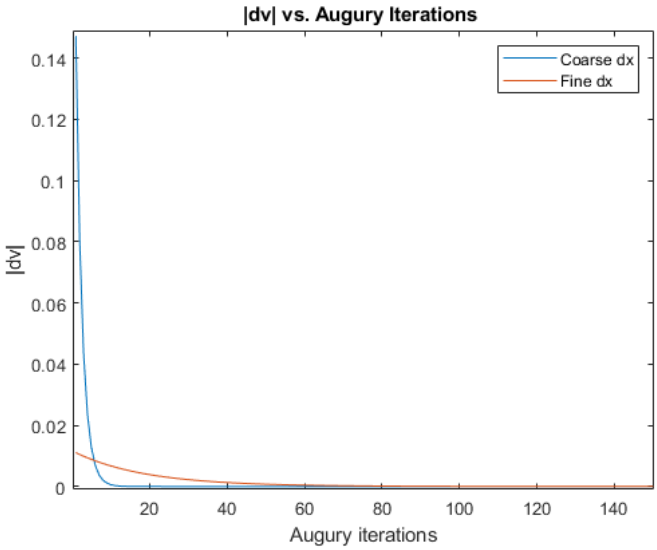


Figure 4.12: **Augury Iterations Convergence** Grid resolution convergence of the augury iterations.

4.5 Method Limitations

The augury iterations will capture the collisions along the normal and remove the collisions and cohesion in the tangential direction. An issue arises with tangential collisions that get filtered specifically when the curvature of the geometry is large in between elements. As a result, penetration of the boundary can occur. One solution to this is to not filter the tangential component of the MPM collision. This conserves the global linear and angular momentum and captures the tangential collision. However, the tangential component will inherit the numerical cohesion of MPM.

Another approach that can be used alongside to prevent the collision is the treatment of collisions via impulses [10]. The impulses are applied iteratively, where convergence can sometimes be slow. Nevertheless, the required number of iterations is reduced since most of the collisions are accounted for by our method. We prove that the impulses also conserve linear and angular momentum in Appendix A.

4.6 Impulses

Another method that is used to resolved collisions of volumetric meshes is the impulse approach [10]. To apply the impulses we can use boundary boxes generated by the surface mesh elements between their t_n position and expected t_{n+1} position. This operation will make a list of all the possible elements that can collide during the time step. Element pairs consisting of point triangle pairs and segment-segment pairs are selected based on proximity first and then on constant collision detection [19] (CCD). Once the elements are selected impulses are applied iteratively. This is done until the magnitude of the applied impulses reaches a tolerance. This method is not guaranteed to prevent penetration, nevertheless is robust enough to be considered the standard for graphics applications. When the impulses are used alongside the MPM transfers The number of iterations required to converge is decreased due to most of the collision being resolved.

4.6.1 Comparasion Resample Augury vs. Impulses

The method is compared with the impulses from [10]. In addition, a combination of the resampling approach used to resolve the collision before applying impulses is shown for explicit forward Euler Fig. 4.13 and implicit backward Euler Fig. 4.14. For explicit time step both methods conserve linear and angular momentum, the impulses method is more energetic Fig. 4.15. This can be attributed to the dissipation of energy from the APIC transfers. The number of iterations required at each substep is reduced from 100 to 10 if we use the transfers to apply an impulse first. This combination also conserves momentum. For the implicit time step Fig. 4.16 we see that the impulses break the conservation of angular momentum, this can be attributed to the dissipation of the implicit time step and the symmetry of the applied impulses from the iterations. We note that the resample method does not break the conservation of angular momentum during the implicit. As the combination of the approaches is used we notice a similar behavior than explicit having faster convergence as most collisions are captured by the APIC transfers.

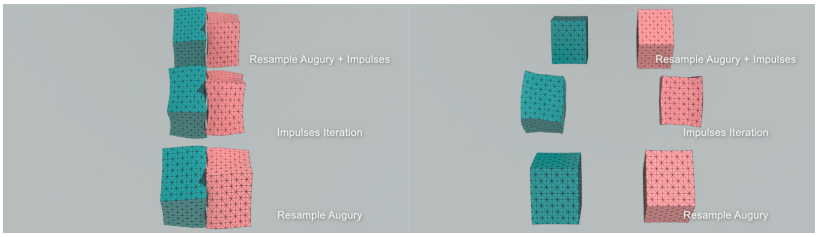


Figure 4.13: **Explicit Collision Approach Comparasion**. Resampling, impulses, and resampling with impulses are shown on two colliding boxes using explicit Euler.

4.7 Fricition

Column friction can be applied similar to [10] in the traditional way. This is done when an elastic impulse is applied. The Coulomb's friction parameter μ is used. The condition for the tangential velocity is seen in Eq.4.60. The friction condition is only enforced when the

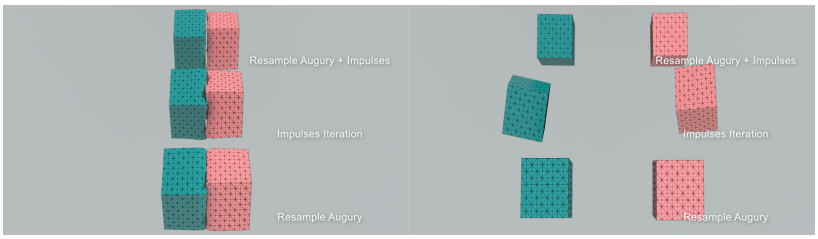


Figure 4.14: **Implicit Collision Approach Comparasion**.Resampling, impulses, and resampling with impulses are shown on two colliding boxes using backward Euler.

change in the velocity comes from the impulses in the normal direction Δv_N.

$$\mathbf{v}_T = max(1 - \mu \frac{\Delta v_N}{|\mathbf{v}_T^{pre}|}, 0)\mathbf{v}_T^{pre} \tag{4.60}$$

The classical friction problem of a sliding block is shown below the method can model the friction and provide a plausible solution. This can be seen in Fig.4.17 where the results match closely to the analytical solutions. Having the block unable to slide where the friction coefficient is slightly larger than the analytical solution.

4.8 Results

4.8.1 Resampling

The advantages of resampling are presented in the simple test of 2 colliding circles Fig.4.18 where the rotation causes the particles to not be aligned at the moment of impact missing the collision. Resampling is used to capture the collision with similar behavior in 2D and 3D. In 3D Fig.4.19 shows the collision being missed in 3D and penetration occurs and in Fig.4.20 resampling is done and the collision is captured accordingly.

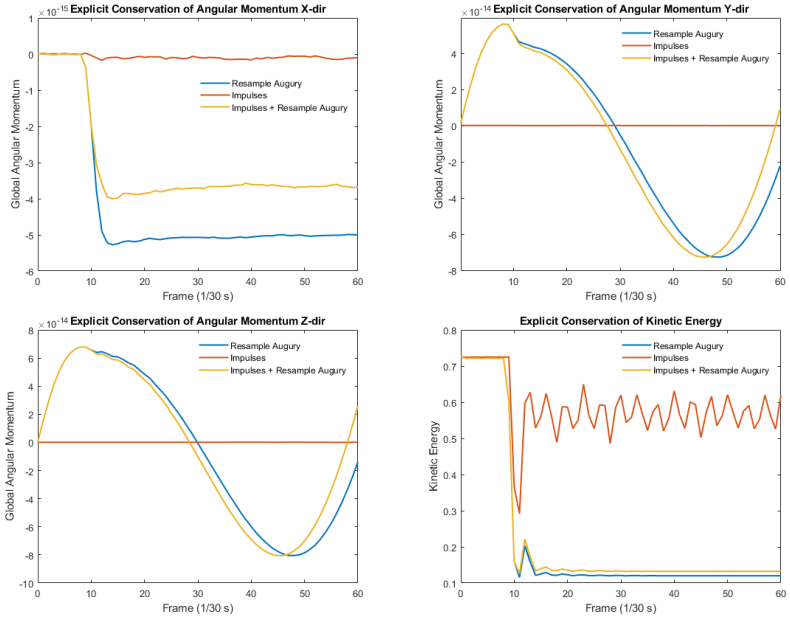


Figure 4.15: **Explicit Conservation of Momentum and Kinetic Energy** Both approaches and applying the resampling approach and then the impulses conserve linear and angular momentum., the Impulses is more energetic

4.8.2 Cohesion Spheres

To demonstrate the severity of numerical cohesion a coarse grid is used to capture the collision between two objects. In 2D, the spheres with no augury iterations exhibit significant numerical cohesion resulting in sticky behavior 4.21. While this cohesion can be reduced by increasing the grid resolution the method can miss the collision. When augury iterations are used we can in figure 4.21 that the cohesion is removed for the collision in the coarse grid. A forced collision is used to demonstrate similar behavior in 3D using a coarse background grid. The typical cohesive behavior resulting in sticking is shown in Fig.4.22. Using Augury to remove the normal cohesion only in Fig .4.24 sticking in the tangential component prevents the spheres to slide. Using Augury to remove the tangential and the normal cohesion in Fig.4.24 the sticking behavior is not present resulting in the spheres sliding.

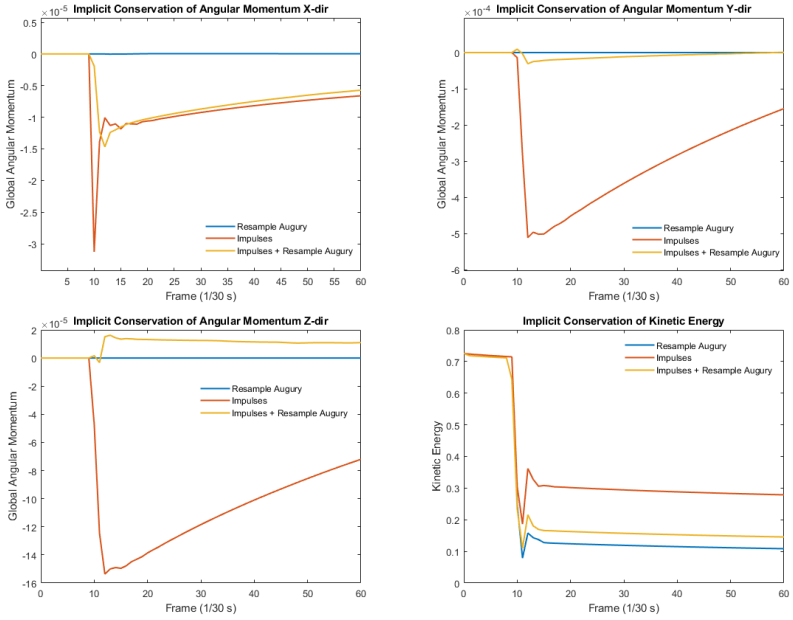


Figure 4.16: **Implicit Conservation of Momentum and Kinetic Energy**. The resampling approach does not break the conservation of angular momentum while the impulses even applied after the resampling break the conservation of angular momentum.

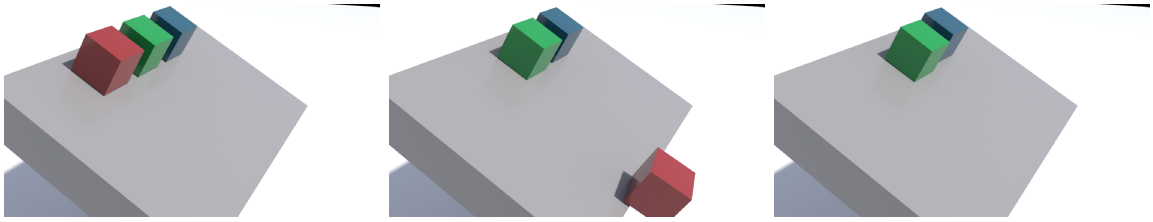


Figure 4.17: **Slidding Block w/ Friction**. A 30 degree incline is used to model a classical friction problem. The **Red block** slides without friction, The **Green block** with $\mu = 0.57$ is able to slide, the **Blue block** with $\mu = 0.58$ the block is unable to slide.

4.8.3 Falling Bunnies

A gravity-driven simulation is used to show the method's capabilities of resolving self-collision, the collision between objects, and collision with boundary conditions. The walls and floor are applied in the MPM grid. In Fig 4.25 the collision from the bunny deforming

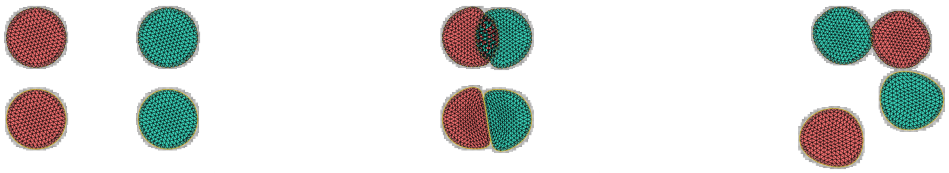


Figure 4.18: **2D Resampling** Collision of two circles with initial linear and angular velocities are presented. At the selected grid resolution resampling is required to capture the collision.

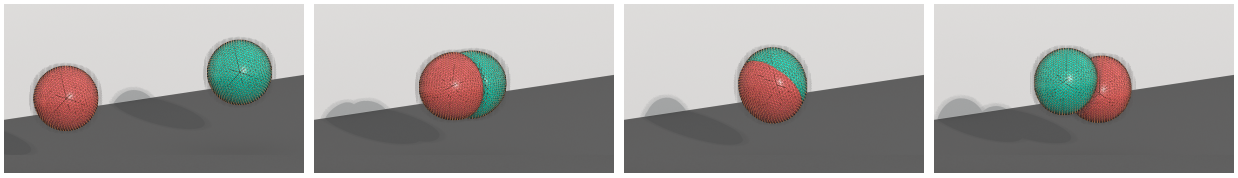


Figure 4.19: **3D Without Resampling** Two spheres are given initial linear and angular velocities. The collision is missed when the simulation uses only the nodes as collision particles.

under gravity is generated folds from self-collision. The collision is all the features including the ears are resolved without penetration.

4.8.4 Twisting Legs

The twisting of a body mesh is used to stress-test the collision capture. The mesh is constrained on the legs and hands and rotated to generate a twist motion. The method uses both the APIC transfers and impulses to resolved the collision from the large deformation. The cohesion is removed and we show the ability to separate by untwisting. As the method performance can be shown in table 4.1 the method has application to dynamics graphics simulations.

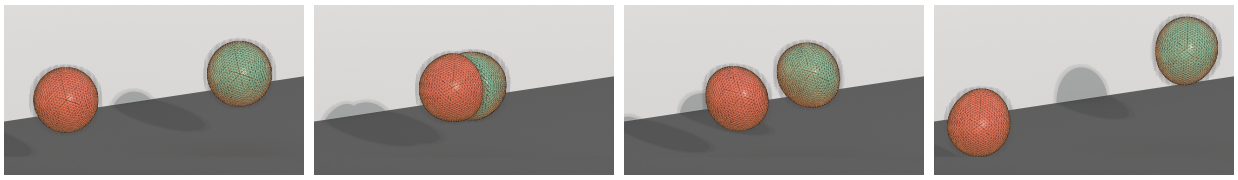


Figure 4.20: **3D Resampling** Two spheres are given initial linear and angular velocities. The collision is captured when the simulation utilizes resampling of the boundary particles.

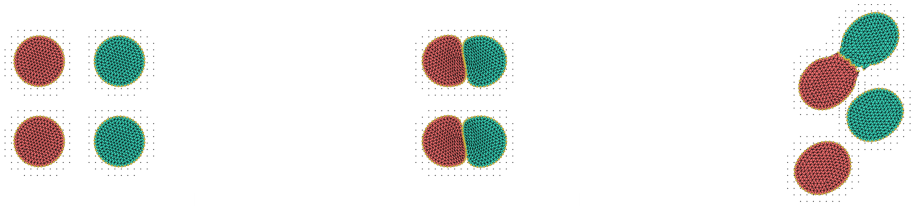


Figure 4.21: **2D Cohesion** The inherited cohesion for coarse grid coming from MPM is removed and the colliding circles are allow to separate.

4.8.5 Roller Test

The method can handle capable of handling large deformations under severe compression. Corotated Elasticity is used for the material. A fine resolution grid allows the method to use alleviate the cohesion in the tangential direction while retaining the dynamics of a stick-slip collision. This has proven effective to simulate a sphere in Fig. 4.27 and the classical Armadilo geometry used in graphics in Fig. 4.28. The rollers are included as kinematic constraints having their velocity splat to the background MPM grid.

4.8.6 Simulation Applications

This algorithm can be leveraged in the simulations with collisions for mesh models with animation purposes. In Fig. 4.29 only augury and resampling are used to capture the collision of opening and closing a mouth. The method can handle the collision without any cohesion. Extensions to incorporate muscles and bones can be made later to approximate the dynamics in a human body. In Fig. 4.30 our method can capture the collision in a moving

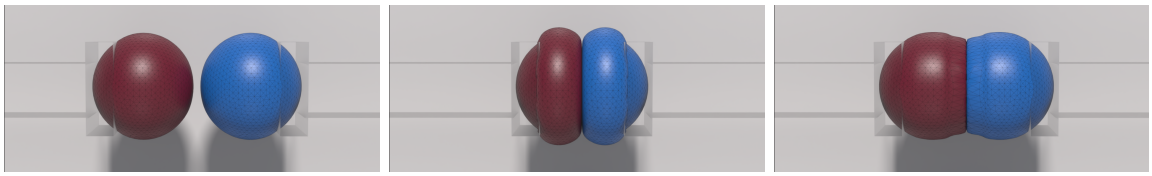


Figure 4.22: **3D Cohesion** The forced collision between 2 spheres in a coarse mesh when the spheres are separated numerical sticking is severe.

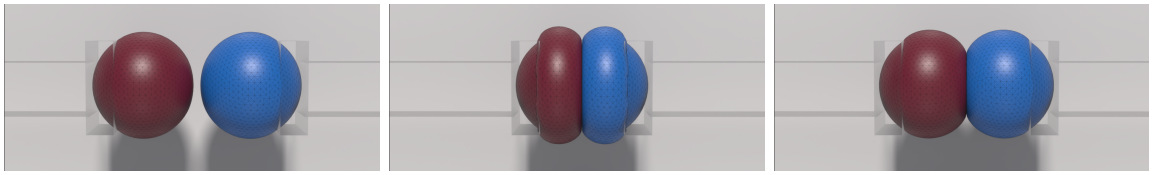


Figure 4.23: **3D w/ Tangential Cohesion** The forced collision between 2 spheres in a coarse mesh when the spheres are separated numerical sticking in the tangential direction locks the spheres in place.

simulation that uses the interior nodes to specify the motion. This collision is centered in the legs, shoulder, and pelvis. More geometrical features such as muscles and bones can be added for a better approximation of body motion.

4.9 Performance

4.10 Limitations and Future Work

An efficient algorithm for collisions that conserves linear and angular momentum has diverse applications in simulating dynamics involving multi-material collisions has been proposed. This method can resolve the collisions of multiple objects and is easily coupled with other MPM methods via the background grid. The main limitations of the method are inherited from MPM. This comes as a loss in the kinetic energy loss when the transfers are used to resolve the collision. In addition, a CFL restriction from the background is needed to guar-

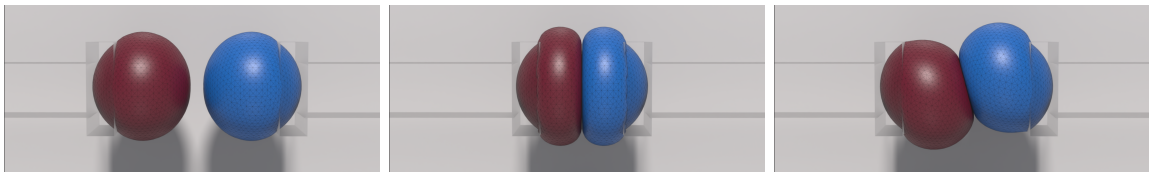


Figure 4.24: **3D No Cohesion** The forced collision between 2 spheres in a coarse mesh when the spheres are separated numerical sticking is removed and the spheres slide from the large forces and deformation.

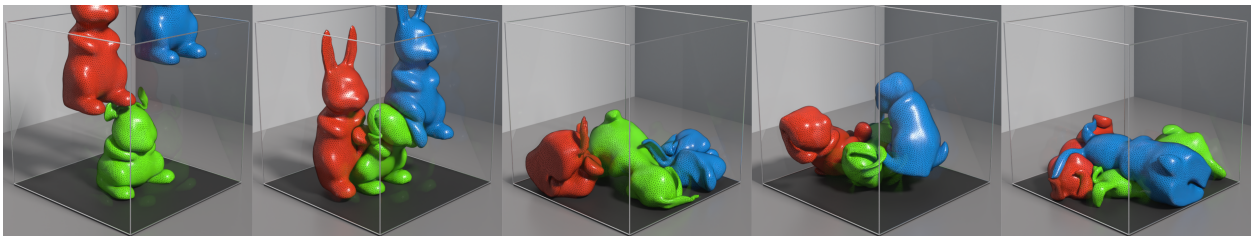


Figure 4.25: **Falling Bunnies** Gravity-driven simulation of multiple bunny tet mesh.

antee that the collision is not missed. The method supports other more energy-conserving transfers, but their advantages have yet to be studied fully for volumetric collision handling. The approach of updating the boundary to propagate the collision information can cause elements to invert under large time steps. Corotated elasticity can handle inversion, however, for other elasticity models, the inversion is not supported.

The tetrahedral elements are suitable for fracture problems, the method can incorporate fracture failure to be used for visual effects and engineering applications. The implicit time step can allow simulation for stiff materials and the conservative linear and angular momentum without the cohesion can be used to capture fracture dynamics including material failure, breaking, and tearing.

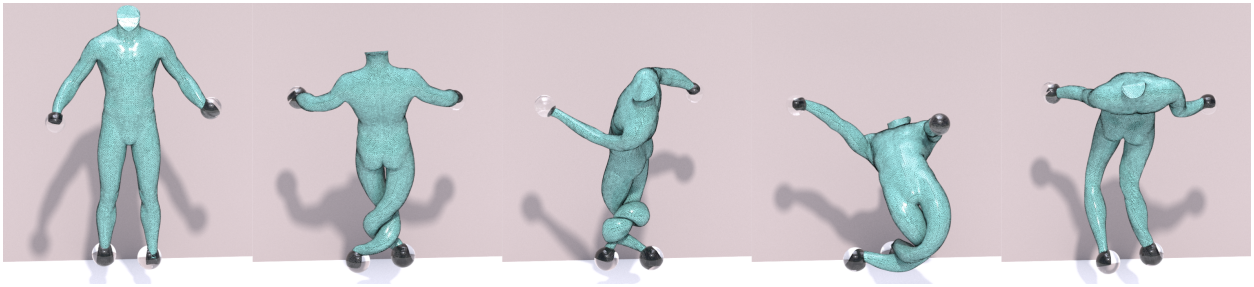


Figure 4.26: **Leg Twist**. The method is tested with large deformations while the legs are twisting. The collision is resolved and the legs manage to twist and untwist.

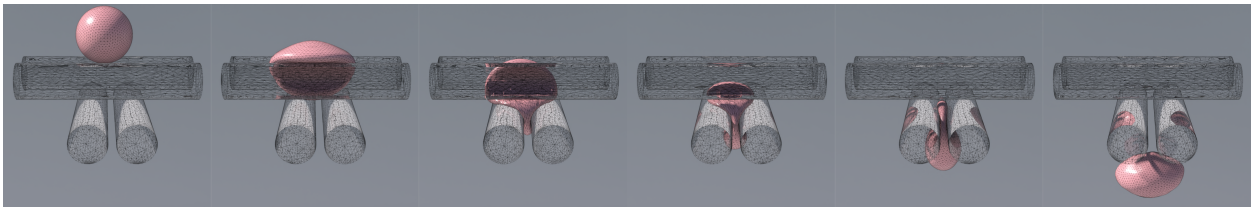


Figure 4.27: **Sphere On Roller**. A sphere under rotating constraints with friction is subjected to large deformations.

Figure 4.28: **Armadillo On Roller**. The classical Armadillo mesh used in graphics is used to test the collision under kinematic boundary conditions.

Figure 4.29: **Lip Collision**. We can capture the collision at the lips opening and closing without cohesion.

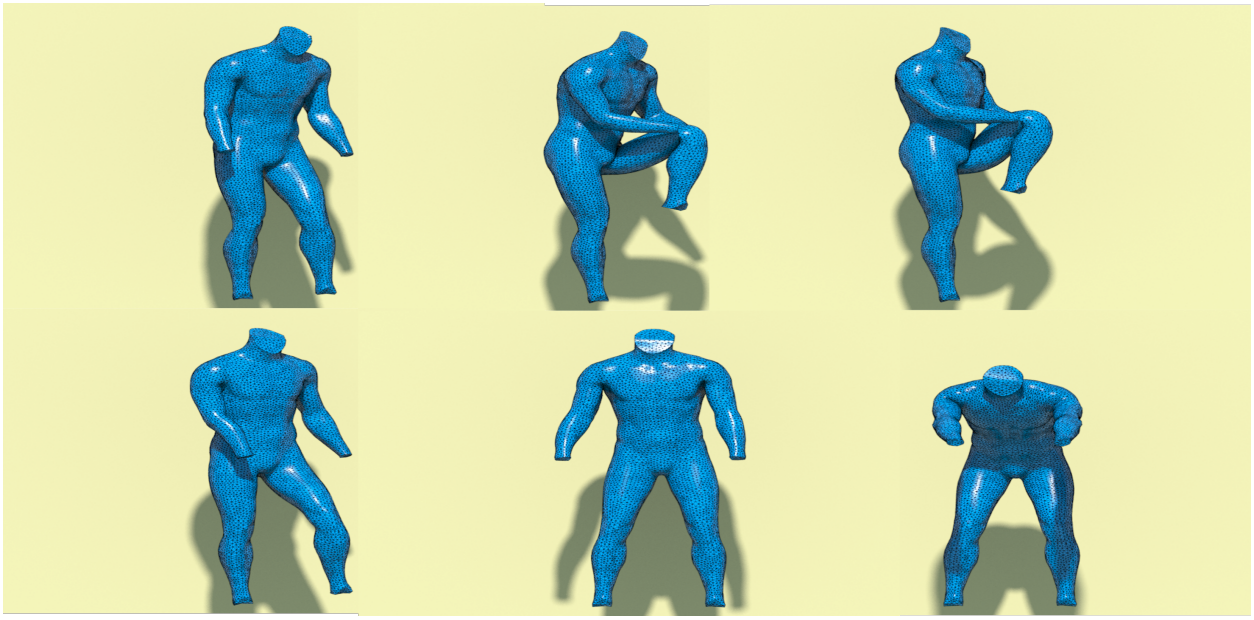


Figure 4.30: **Moving Body** The inner nodes drive a streching motion that produces collisions in the surface.

Example	Avg frame (s)	# Nodes	E, ρ	Resample/Impulses
Bunnies (Fig. 4.25)	175	210k	750, 100	R
Armadillo (Fig. 4.28)	150	68k	5000, 1000	R+I
Lips (Fig. 4.29)	60	431k	100, 100	R
Moving Body (Fig. 4.30)	90	50k	5000, 25	R+I
Leg Twist (Fig. 4.26)	210	145k	7000, 1000	R+I

Table 4.1: Average time per frame (in secs) for each of the 3D examples shown. Examples were run on an intel 12-core CPUs 3.5GHz.

CHAPTER 5

Surface Tension Formulations for the Material Point Method

Surface tension effects are indispensable tools in modern computer graphics applications. Whether, Surface tension driven flows like those in milk crowns [20], droplet coalescence [21, 22, 23, 24, 25] and bubble formation [26, 27, 28] comprise some of the most visually compelling fluid motions. However, materials with high surface tensions have not been explored to a great extent. For example, mercury and other liquid metals exhibit high surface tensions and yet remarkably have small viscosities [29]. An implicit MPM for surface tension can model liquid metals among other types of material and fluids. Fluid simulations difficulties arise from the governing Navier Stokes equations, where an infeasible resolution and time step are required to capture the fluid behavior. Explicit MPM has been used to model compressible fluid-structure interactions [7], however, an implicit formulation for compressible fluids has not been implemented. Implicit methods often require more computations but are more stable than explicit methods. This stability allows the implicit methods to take larger time steps than explicit methods. Thus, it can be shown that for larger scales this increment in time step results in faster simulation times and more efficient codes that require less computational cost [30]. The implicit and explicit MPM formulation has been derived and validated for some elasticity problems and can be extended to surface tension. Surface tension has a strong correlation to the dynamics of fluids. Appropriate modeling of the surface tension is required to model water, liquid metals, and melting solids. Extending MPM to model surface tension can take advantage of the inherent multiphysics the method

can handle. Surface tension forces are derived from an energy-based formulation. The spatial variation in the surface forces can be characterized in terms of the potential energy Ψ^s associated with surface tension:

$$\Psi^s = \int_\Gamma k^\sigma(\mathbf{x}) ds(\mathbf{x}). \tag{5.1}$$

Here the surface tension coefficient k^σ is proportionate to the relative cohesion and adhesion at the interface between the two fluids.

We demonstrate the effectiveness of the method in modeling large stiffness materials with large surface tensions, in addition, to being linear and angular momentum conserving.
Our contributions can be summarized as:

- An updated Lagrangian discretization of surface tension forces defined as the gradient of the surface energy concerning the flow over a time step.
- A particle/level set based boundary particle quadrature rule for computing the surface area of a collection of discrete particles.
- A momentum-conserving particle resampling technique for particles near the surface tension liquid interface.

5.1 Related Work

Surface tracking and Lagrangian meshes: Surface tracking techniques that resolve the liquid surface with explicit mesh topology have many advantages for surface tension effects since they accurately allow for curvature estimation and boundary application [22, 23, 31]. Brochu et al. [32] use the surface tracking formulation of Brochu and Bridson [33] to explicitly track topological changes of the liquid interface. This allows them to accurately resolve surface tension boundary conditions with a cut-cell Voronoi discretization based on the embedded tetrahedron formulation of Batty et al. [34]. Sun et al. [35] and De Goes et al. [36]

use similar Voronoi-based discretizations. While the previously mentioned techniques use an accompanying volumetric, usually Eulerian, discretization, Da et al. [37] use a boundary element formulation with fluid represented by a tracked triangulation of its boundary. They use the surface tracking formulation of Brochu and Bridson [33] to resolve merging and pinching behaviors. Thürey et al. [21] also use a surface tracking approach and build on the formulation of Sussman and Ohta [38] and implicit mean curvature flow approach of Eckstein et. al. [39]. Misztal et al. [40] also define the surface tension force as the gradient of the surface area. They build on [41] to develop an implicit approach for tetrahedron-based incompressible flows with surface tension. Wicke et al. [42] use dynamic topology tetrahedron meshes. Zhu et al. [26] use dynamic codimensional simplicial meshes to resolve thin sheets, filaments and droplets. They use the surface tension force approach of Zheng et al. [20] combined with a novel rim-based surface tension on the boundary of thin sheets.

Particle based methods: Brackbill et al. [43] developed the Continuum Surface Force (CSF) approach for PIC which defines normals and curvatures as gradients of color functions (defined on the grid after transfer from particles). The surface tension force is effectively regularized as color function transitions over a few cell widths. Muller [44] et al. also use the gradient of a color function in a per-particle manner based on the work of Morris [45], which is a generalization of the CSF model of Brackbill et al. [43] to SPH. CSF has a number of drawbacks, including that: normalization of the color gradient is noisy for internal particles, curvature estimation is very sensitive to particle sampling uniformity and that CSF forces are not exactly conservative. [46, 47, 48]. Yu et al. [49] use surface tracking with SPH to define surface tension forces. In general, the determination of which particles should be considered to be on the boundary of a particle-based domain is an open problem [50, 51, 52, 53, 54, 55]. Orthmann et al. [56] use a discrete delta function approach with SPH to define particle-based surface area. Mueller et al. combine SPH with surface tracking [31].

Implicit Surface Tension: Popinet [57] provides a useful review of implicit surface tension techniques. Bänsch et al. [58] use a semi-implicit approach akin to one step of Newton iteration with an explicit mesh, FEM discretization of a surface energy-based formulation. Hysing [59] uses a semi-implicit approach based on a variational CSF. Sussman and Ohta [38] also use a semi-implicit approach based on mean curvature flow. Although these approaches are not fully implicit, Popinet [57] shows they are equivalent to the addition of a surface viscosity that damps capillary waves and leads to a $O(\Delta x)$ time step. Hochstein and Williams [60] developed one of the first implicit approaches for surface tension between two phases. Hou et al. [61] add and subtract a Laplacian term as an approximation to the surface tension Hessian in a boundary integral formulation. Jaruta et al. [62] use a Lagrangian formulation of incompressible flow and treat surface tension in a fully implicit manner. Zheng et al. [20] develop a hybrid particle/grid based implicit technique for surface tension with a three-dimensional version of Schroeder et al. [63] using a triangle mesh to discretize the surface tension force.

5.2 Governing Equations

The governing equations for the physical system are obtained from the conservation of mass and momentum, the Eulerian form will be:

$$\begin{aligned} \rho \tfrac{D\mathbf{v}}{Dt} &= \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} \\ \tfrac{D\rho}{Dt} &= -\rho \nabla \cdot \mathbf{v}, \mathbf{x} \in \Omega^t. \end{aligned} \tag{5.2}$$

Where ρ is the Eulerian mass density, $\mathbf{v}$ is the Eulerian material velocity, $\boldsymbol{\sigma}$ is the Cauchy stress and $\mathbf{g}$ is the gravitational acceleration. Boundary conditions for these equations are associated with a free surface for solid material, surface tension for liquids, and/or prescribed velocity conditions. We use $\partial\Omega^t_N$ to denote the portion of the time t boundary subject to free surface or surface tension conditions and $\partial\Omega^t_D$ to denote the portion of the boundary

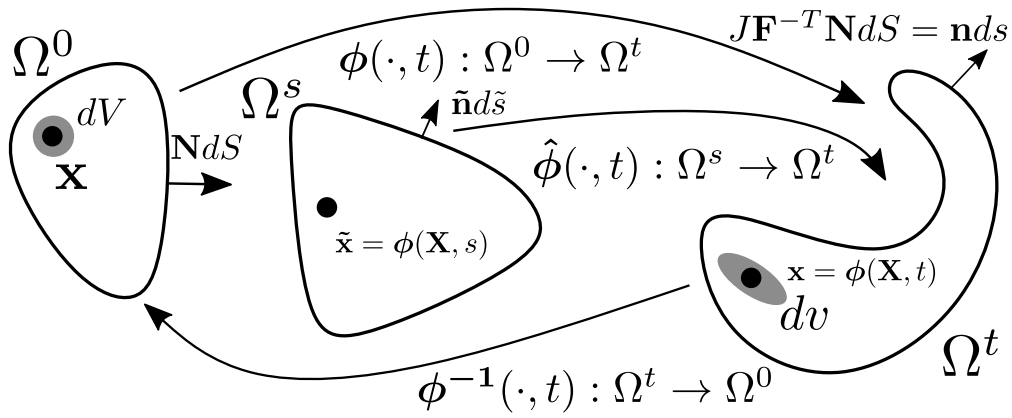


Figure 5.1: **Material Flow Map**. A flow map depiction of the way we mathematically model material deformation over time

with Dirichlet velocity boundary conditions. Free surface conditions and surface tension boundary conditions are expressed as:

$$\mathbf{t} = \mathbf{n} \cdot \boldsymbol{\sigma}, \mathbf{x} \in \partial\Omega_N^t \tag{5.3}$$

where $\mathbf{t} = \mathbf{0}$ for free surface conditions and $\mathbf{t} = k^\sigma \kappa \mathbf{n} + \nabla^S k^\sigma$ for surface tension conditions. With prescribed velocity $\mathbf{v} \cdot \mathbf{n} = v_{\text{bc}}^n,\ \mathbf{x} \in \partial\Omega_D^t$ (see Figure 5.1). κ is the mean curvature, k^σ is the coefficient of surface tension [64], p is the pressure in the fluid and $\mathbf{g}$ is the gravitational acceleration.

5.2.0.1 Constitutive Models

Each material point is treated as a liquid, the Cauchy stress $\boldsymbol{\sigma}$ is defined in terms of pressure and viscous stress:

$$\boldsymbol{\sigma} = -p\mathbf{I} + \mu\left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}^T\right),\ p = -\frac{\partial \psi^p}{\partial J},$$

with $\psi^p(J) = \frac{\lambda^l}{2}(J-1)^2$. Here λ is the bulk modulus of the liquid and μ is its viscosity.

The constitutive model penalizes the compressibility of the liquid. Requirering the use of the constitutive relation $p = -k^p(j-1)$ where k^p is the bulk modulus and $j(\mathbf{x},t) =$

$J(\boldsymbol{\phi}^{-1}(\mathbf{x},t),t)$ is the Eulerian deformation gradient determinant. Intuitively, larger values of the bulk modulus penalize compressible flow ($J \neq 1$) more severely.

5.3 Potential Energy

The discretization of the momentum balance in Equation (5.2) is based on the potential energy of the material. The potential energy associated with surface tension is proportionate to the surface area of the material as it evolves in the flow [64]. Using $\Psi^s(\boldsymbol{\phi}(\cdot,t))$ to denote the total surface tension potential energy at time t under the flow map

$$\Psi^s(\boldsymbol{\phi}(\cdot,t)) = k^\sigma \int_{\partial\Omega^t} ds(\mathbf{x}) = k^\sigma \int_{\partial\Omega^s} |\hat{J}\hat{\mathbf{F}}^{-T}\tilde{\mathbf{n}}| ds(\tilde{\mathbf{x}}). \tag{5.4}$$

Here k^σ is the coefficient of surface tension. Increasing values of the surface tension coefficient correspond to materials with high surface energies like water drops at small scales or liquid metals. In Equation (5.4) the integral over $\partial\Omega^t$ can be written as one over $\partial\Omega^s$ using the surface integral change of variables, where $\tilde{\mathbf{n}}$ is the outward unit normal to the initial material boundary (see Figure 5.1). Here $|\hat{J}\hat{\mathbf{F}}^{-T}\tilde{\mathbf{n}}|$ can be shown to be the ratio of infinitesimal surface areas in the current (ds) and time s configurations ($d\tilde{s}$). The pressure and gravitational forces in the liquid can also be defined from their associated potential energies.

$$\Psi^p(\boldsymbol{\phi}(\cdot,t)) = \int_{\Omega^0} \frac{k^p}{2}(J-1)^2 d\mathbf{X},\ \ \Psi^g(\boldsymbol{\phi}(\cdot,t)) = \int_{\Omega^0} \boldsymbol{\phi}\cdot RJ\mathbf{g} d\mathbf{X}. \tag{5.5}$$

For the pressure and surface tension potential energies, we define the following potential energy densities.

$$\hat{\Psi}^p(J) = \frac{k^p}{2}(J-1)^2, \quad \hat{\Psi}^s(\hat{\mathbf{F}}, d\mathbf{A}) = k^\sigma |\hat{J}\mathbf{z}\hat{\mathbf{F}}^{-T} d\mathbf{A}|. \tag{5.6}$$

The momentum balance in Equation (5.2), including the surface tension and velocity boundary conditions, is equivalent to the variational form.

$$\int_{\Omega^t} \rho w_\alpha \frac{Dv_\alpha}{Dt} d\mathbf{x} = -\frac{d}{d\epsilon}\mathrm{PE}(0;\mathbf{w}) - \mu \int_{\Omega^t} \epsilon^v_{\alpha\beta}\epsilon^w_{\alpha\beta} d\mathbf{x}, \quad \begin{array}{l} \forall \mathbf{w} : \Omega^t \to \mathbb{R}^d \\ \mathbf{w}\cdot\mathbf{n} = 0, \mathbf{x} \in \partial\Omega^t_D \end{array} \tag{5.7}$$

where $\text{PE}(\epsilon;\mathbf{w}) = \Psi^s(\hat{\boldsymbol{\phi}}(\cdot,t)+\epsilon\hat{\mathbf{w}})+\Psi^p(\boldsymbol{\phi}(\cdot,t)+\epsilon\mathbf{W})+\Psi^g(\boldsymbol{\phi}(\cdot,t)+\epsilon\mathbf{W})$ for $\hat{\mathbf{w}}(\tilde{\mathbf{x}}) = \mathbf{w}(\hat{\boldsymbol{\phi}}(\tilde{\mathbf{x}},t))$ and $\mathbf{W}(\mathbf{X}) = \mathbf{w}(\boldsymbol{\phi}(\mathbf{X},t))$. This is obtained by taking the dot product of the momentum balance with w_α and integrating over Ω^t, as well as integration by parts. The left hand side of Equation (5.7) can be written in the updated Lagrangian view by changing variables to Ω^s with $s < t$ resulting in

$$\int_{\Omega^t} \rho w_\alpha \frac{Dv_\alpha}{Dt} d\mathbf{x} = \int_{\Omega^s} \hat{\rho}\hat{w}_\alpha \frac{\partial \hat{v}_\alpha}{\partial t} \hat{J} d\tilde{\mathbf{x}} \tag{5.8}$$

where $\hat{\rho}(\tilde{\mathbf{x}},t) = \rho(\hat{\boldsymbol{\phi}}(\tilde{\mathbf{x}},t),t)$.

Here $\epsilon^w = \frac{1}{2}\left(\frac{\partial w_\alpha}{\partial x_\beta} + \frac{\partial w_\beta}{\partial x_\alpha}\right)$ and $\epsilon^v = \frac{1}{2}\left(\frac{\partial v_\alpha}{\partial x_\beta} + \frac{\partial v_\beta}{\partial x_\alpha}\right)$. We can change the change variables in the viscosity term as follows:

$$\mu \int_{\Omega^t} \epsilon^v_{\alpha\beta}\epsilon^w_{\alpha\beta} d\mathbf{x} = \mu^l \int_{\Omega^s} \hat{\epsilon}^v_{\alpha\beta}\hat{\epsilon}^w_{\alpha\beta} \hat{J} d\tilde{\mathbf{x}} \tag{5.9}$$

5.4 Discretization

The governing equations are discretized with MPM and APIC [65, 66, 3]. To discretize in space at time t^n particle samples $\mathbf{x}_p^n$ of the domain Ω^{t^n} are used. Each particle approximates the deformation gradient determinant J_p^n, the velocity in terms of constant $\mathbf{v}_p^n$ and affine velocity $\mathbf{A}_p^n$. The initial mass $m_p = \rho(\mathbf{x}_p^0, 0)V_p^0$ and volume V_p^0 are also stored. Conservation of mass is applied at the particles providing the volume from the deformation gradient determinant as $V_p^n = J_p^n V_p^0$.

Two approaches can discretize the surface tension forces. These approaches consist of adding massless particles in the boundary or resampling mass into the boundary.

5.4.1 Massless Particles Approach

MPM is extended by adding massless particles $\mathbf{x}_q^n$ with boundary area samples $d\mathbf{A}_q^n$ for surface tension energy quadrature (see Figure 5.2. The massless particles are added at the beginning and removed at the end of each time step. The massless particles do not affect

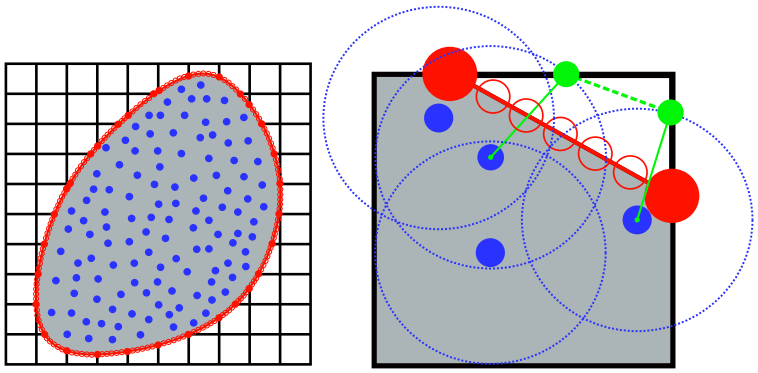


Figure 5.2: **Isocontour**. *(Left)* MPM fluid particles (dark blue) give rise to a level set isocontour (red). *(Right)* The isocontour is of the level set formed by the union of each fluid particle's spherical level set. Surface tension forces are valuated at sample points (open red circles) placed on the isocontour face.

the conservation of mass. However, the conservation of momentum changes from the surface tension force they generate.

First transfer the time t^n particle mass and momentum to the grid using APIC transfers [3] with quadratic B-spline interpolating functions $N_{\mathbf{i}}(\mathbf{x}) = N(\mathbf{x} - \mathbf{x}_{\mathbf{i}})$ defined on the Eulerian grid nodes $\mathbf{x}_{\mathbf{i}}$ as

$$m_{\mathbf{i}}^n = \sum_p m_p N_{\mathbf{i}}(\mathbf{x}_p^n), \quad m_{\mathbf{i}}^n \mathbf{v}_{\mathbf{i}}^n = \sum_p m_p N_{\mathbf{i}}(\mathbf{x}_p^n) \left(\mathbf{v}_p^n + \mathbf{A}_p^n (\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n) \right), \tag{5.10}$$

The grid momentum is updated by discretizing the right-hand side of Equation (5.8) with $s = t^n$ as

$$\int_{\Omega^s} \hat{\rho} \hat{w}_\alpha \frac{\partial \hat{v}_\alpha}{\partial t} \hat{J} d\tilde{\mathbf{x}} \approx \int_{\Omega^{t^n}} \hat{\rho} \hat{w}_\alpha \frac{\hat{v}_\alpha(\tilde{\mathbf{x}}, t^{n+1}) - v_\alpha(\tilde{\mathbf{x}}, t^n)}{\Delta t} \hat{J} d\tilde{\mathbf{x}}. \tag{5.11}$$

We use the backward difference in time. To discretize in space, v_α, $\hat{v}_\alpha$ and $\hat{w}_\alpha$ are approximated using the quadratic B-splines as $v_\alpha(\tilde{\mathbf{x}}, t^n) = v_{\mathbf{i}\alpha}^n N_{\mathbf{i}}(\tilde{\mathbf{x}})$, $\hat{v}_\alpha(\tilde{\mathbf{x}}, t^{n+1}) = \hat{v}_{\mathbf{i}\alpha}^{n+1} N_{\mathbf{i}}(\tilde{\mathbf{x}})$ and $\hat{w}_\alpha = \delta_{\beta\alpha} N_{\mathbf{i}}(\tilde{\mathbf{x}})$ respectively. The integral 5.11 then becomes

$$\left(\int_{\Omega^{t^n}} \hat{\rho} \hat{J} N_{\mathbf{i}} N_{\mathbf{j}} d\tilde{\mathbf{x}} \right) \frac{\hat{v}_{\mathbf{i}\beta}^{n+1} - v_{\mathbf{i}\beta}^n}{\Delta t}. \tag{5.12}$$

The integral is approximated using the MPM particles as quadrature points

$$\int_{\Omega^{t^n}} \hat{\rho}\hat{J}N_{\mathbf{i}}N_{\mathbf{j}}d\tilde{\mathbf{x}} \approx \sum_p \hat{\rho}(\mathbf{x}_p^n, t^{n+1})\hat{J}(\mathbf{x}_p^n, t^{n+1})V_p^n N_{\mathbf{i}}(\mathbf{x}_p^n)N_{\mathbf{j}}(\mathbf{x}_p^n) = \sum_p m_p N_{\mathbf{i}}N_{\mathbf{j}} \tag{5.13}$$

where R is the Lagrangian mass density defined by $R(\mathbf{X}_p, t) = \rho(\boldsymbol{\phi}(\mathbf{X}_p, t), t)$, and $\mathbf{X}_p = \boldsymbol{\phi}^{-1}(\mathbf{x}_p^n, t^n) = \mathbf{x}_p^0$. If this integral is replaced with the lumped mass approximation the mass transfer becomes 5.10.

To update the grid momentum the total potential energy is discretized using the MPM points for quadrature

$$e(\hat{\mathbf{x}}) = \sum_p \hat{\Psi}^p \left(J_p^{n+1}(\hat{\mathbf{x}})\right) V_p^n + \sum_q \hat{\Psi}^s \left(\hat{\mathbf{F}}_q\left(\hat{\mathbf{x}}\right), d\mathbf{A}_q\right) \approx \mathrm{PE}(0; \mathbf{w}). \tag{5.14}$$

Massless surface tension particles are used in this discretization. using $\hat{\mathbf{x}}$ to denote the vector of all potentially moved grid node positions $\hat{\mathbf{x}}_{\mathbf{i}}$ where the functions $J_p^{n+1}(\hat{\mathbf{x}})$, $\hat{F}_q(\hat{\mathbf{x}})$ are given by

$$J_p^{n+1}(\hat{\mathbf{x}}) = \left(1 + (\hat{\mathbf{x}}_{\mathbf{i}} - \mathbf{x}_{\mathbf{i}}) \cdot \frac{\partial N_{\mathbf{i}}}{\partial \mathbf{x}}(\mathbf{x}_p^n)\right) J_p^n, \quad \hat{\mathbf{F}}_q(\hat{\mathbf{x}}) = \hat{\mathbf{x}}_{\mathbf{i}} \frac{\partial N_{\mathbf{i}}^T}{\partial \mathbf{x}}(\mathbf{x}_q^n). \tag{5.15}$$

The force on grid node $\mathbf{i}$ is then $\mathbf{f}_{\mathbf{i}}(\hat{\mathbf{x}}) = -\frac{\partial e}{\partial \hat{\mathbf{x}}_{\mathbf{i}}}(\hat{\mathbf{x}})$ and the update for the grid momentum is then

$$m_{\mathbf{i}}^n \frac{\hat{\mathbf{v}}_{\mathbf{i}}^{n+1} - \mathbf{v}_{\mathbf{i}}^n}{\Delta t} = \mathbf{f}_{\mathbf{i}}(\mathbf{x} + \Delta t \hat{\mathbf{q}}) + m_{\mathbf{i}}^n \mathbf{g}, \tag{5.16}$$

where $\hat{\mathbf{q}} = \mathbf{0}$ for explicit time integration and $\hat{\mathbf{q}} = \hat{\mathbf{v}}^{n+1}$ for backward Euler time integration. As with $\hat{\mathbf{x}}$, the vector $\hat{\mathbf{v}}^{n+1}$ contains the corresponding velocity vectors defined on each grid node. Furthermore, $\mathbf{x}$ is the vector of all unmoved grid node locations $\mathbf{x}_{\mathbf{i}}$. The α component of the force on grid node $\mathbf{i}$ is then

$$f_{\mathbf{i}\alpha}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial \hat{\Psi}^p}{\partial J}(J_p^{n+1}(\hat{\mathbf{x}}))\frac{\partial N_{\mathbf{i}}}{\partial x_\alpha}(\mathbf{x}_p^n)J_p^n V_0^n - \sum_q \frac{\partial \hat{\Psi}^s}{\partial \hat{F}_{\alpha\gamma}}(\hat{\mathbf{F}}_q(\hat{\mathbf{x}}), d\mathbf{A}_q)\frac{\partial N_{\mathbf{i}}}{\partial x_\gamma}(\mathbf{x}_q^n) \tag{5.17}$$

After the grid update, APIC is used to transfer from grid to particles

$$\mathbf{v}_p^{n+1} = \sum_{\mathbf{i}} N_{\mathbf{i}}(\mathbf{x}_p^n)\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}, \quad A_p^{n+1} = \frac{4}{\Delta x^2}\sum_{\mathbf{i}} N_{\mathbf{i}}(\mathbf{x}_p^n)\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n)^T \tag{5.18}$$

The particles are then updated to their time t^{n+1} positions via $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$.

5.4.2 Mass Resampling Approach

The domain Ω^{t^n} at time t^n is sampled using material points $\mathbf{x}_p^n$. These points also store approximations of the deformation gradient determinant J_p^n, constant velocity $\mathbf{v}_p^n$, affine velocity $\mathbf{A}_p^n$, volume V_p^0, and mass $m_p = \rho(\mathbf{x}_p^0, t^0)V_p^0$. To advance our state to time t^{n+1}, we use the following steps:

1. Resample particle boundary for surface tension.
2. P2G: Conservative transfer of momentum from particles to grid.
3. Update of grid momentum.
4. G2P: Conservative transfer of momentum from grid to particles.

5.4.3 Conservative Surface Particle Resampling

We follow Hyde et al. [67] and introduce special particles to cover the boundary to serve as quadrature points for these integrals. Massless particles easily allow for momentum conserving transfers from particles to the grid and vice versa; however, they can lead to loss of conservation in the grid momentum update step. This occurs when there is a grid cell containing only massless particles. In this case, there are grid nodes with no mass that receive surface tension forces. These force components are then effectively thrown out since only grid nodes with mass will affect the end of time step particle momentum state.

We resolve this issue by assigning mass to each of the surface particles. However, to conserve total mass, some mass must be subtracted from interior MPM particles. Furthermore, changing the mass of existing particles also changes their momentum, which may lead to a violation of conservation. To conserve mass, linear momentum, and angular momentum, we introduce a new particle for each surface particle. We call these balance particles, and like surface particles, they are temporary and will be removed at the end of the time step. We show that the introduction of these balance particles naturally allows for conservation both

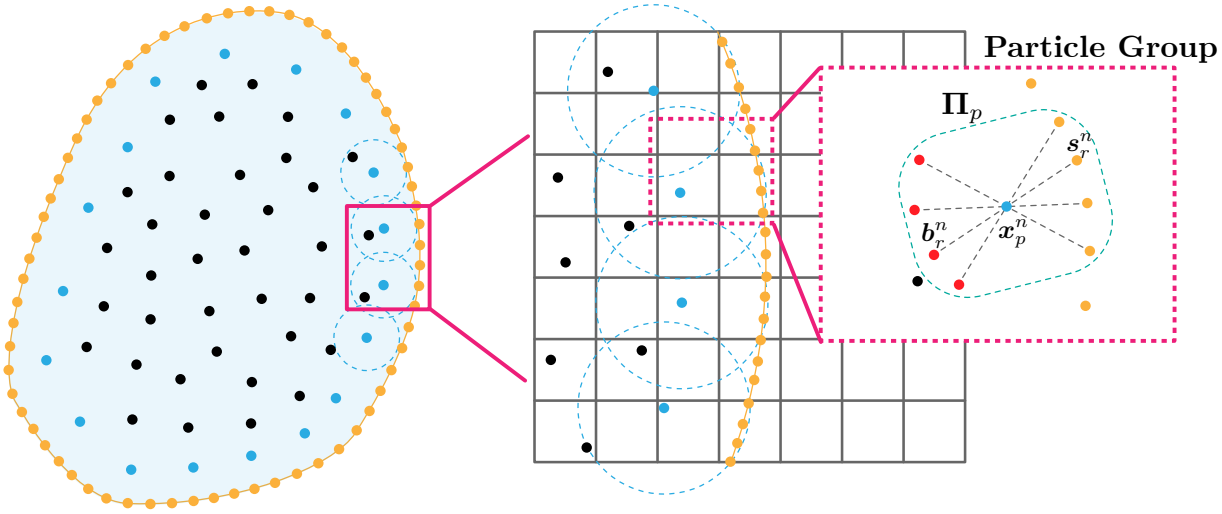


Figure 5.3: **Resampling**. A portion of an MPM fluid in the simulation domain. Surface particles (yellow) are sampled on faces of the zero isocontour of the level set formed by unioning spherical level sets around each MPM particle. Each surface particle generates an associated balance particle (red) such that the closest MPM particle (blue) to a boundary particle lies on the midpoint of a line segment between the surface particle and the balance particle. A single blue particle at $\boldsymbol{x}_p$ may be paired with multiple surface particles and balance particles, and they are considered to be in a particle group $\mathbf{\Pi}_p$. MPM particles that are not associated with any surface tension particles are marked as black.

when they are created at the beginning of the time step and when they are removed at the end of the time step.

5.4.3.1 Surface Particle Sampling

We first introduce surface particles using the approach in [67]. A level set enclosing the interior MPM particles is defined as the union of spherical level sets defined around each interior MPM particle. We compute the zero isocontour of the level set using marching cubes [68] and randomly sample surface particles along with this explicit representation. We employ a strategy of per-triangle Monte Carlo sampling using a robust Poisson distribution,

as described in Figure 1 of [69] (not their blue noise algorithm); uniform triangle sample points are generated as in [70]. We found that this gave better coverage of the boundary without generating particles in a biased, mesh-dependent fashion (see Figure 5.4). We note that radii for the particle level sets are taken to be $0.73\Delta x$ (slightly larger than $\frac{\sqrt{2}}{2}\Delta x$) in 2D and $0.867\Delta x$ (slightly larger than $\frac{\sqrt{3}}{2}\Delta x$) in 3D, where Δx is the MPM grid spacing. This guarantees that even a single particle in isolation will always generate a level set zero isocontour that intersects the grid and will therefore always generate boundary sample points. Note also that as in [67], we use the explicit marching cubes mesh of the zero isocontour to easily and accurately generate samples of area-weighted normals $d\mathbf{A}_r^n$ where $\sum |d\mathbf{A}_r^n| \approx \int_{\partial\Omega^{\iota^n}} d\mathbf{s}$ are chosen with direction from the triangle normal and magnitude based on the number of samples in a given triangle and the triangle area.

5.4.3.2 Balance Particle Sampling

For each surface particle $\mathbf{s}_r^n$, we additionally generate a balance particle $\mathbf{b}_r^n$. First, we compute the closest interior MPM particle for each surface particle $\mathbf{x}_{p(\mathbf{s}_r^n)}^n$. Then we introduce the corresponding balance particle as

$$\mathbf{b}_r^n = \mathbf{s}_r^n + 2\left(\mathbf{x}_{p(\mathbf{s}_r^n)}^n - \mathbf{s}_r^n\right). \tag{5.19}$$

5.4.3.3 Mass and Momentum Splitting

After introducing the surface $\mathbf{s}_r^n$ and balance $\mathbf{b}_r^n$ particles, we assign them mass and momentum (see Figure 5.5). To achieve this in a conservative manner, we first partition the surface particles into particle groups Π_p defined as the set of surface particle indices r such that $\mathbf{x}_p^n$ is the closest interior MPM particle to $\mathbf{s}_r^n$ (see Figure 5.3). We assign the mass m_p of the particle $\mathbf{x}_p^n$ to the collection of $\mathbf{x}_p^n$, $\mathbf{s}_r^n$ and $\mathbf{b}_r^n$ for $r \in \Pi_p$ uniformly by defining a mass of $\tilde{m}_p = \frac{m_p}{2|\Pi_p|+1}$ to each surface and balance point as well as to $\mathbf{x}_p^n$. Here $|\Pi_p|$ is the number of elements in the set. This operation is effectively a split of the original particle $\mathbf{x}_p^n$ with mass

m_p into a new collection of particles $\mathbf{x}_p^n, \mathbf{s}_r^n, \mathbf{b}_r^n,\ r \in \Pi_p$ with masses $\tilde{m}_p$. This split trivially conserves the mass. Importantly, by construction of the balance particles (Equation (5.19)) we ensure that the center of mass of the collection is equal to the original particle $\mathbf{x}_p^n$:

$$\frac{1}{m_p}\left(\tilde{m}_p\mathbf{x}_p^n + \sum_{r\in\Pi_p}\tilde{m}_p\mathbf{s}_r^n + \tilde{m}_p\mathbf{b}_r^n\right) = \mathbf{x}_p^n. \tag{5.20}$$

With this particle distribution, conservation of linear and angular momentum can be achieved by simply assigning each new particle in the collection the velocity $\mathbf{v}_p^n$ and affine velocity $\mathbf{A}_p^n$ of the original particle $\mathbf{x}_p^n$. We note that the conservation of the center of mass (Equation (5.20)) is essential for this simple constant velocity split to conserve linear and angular momentum.

5.4.4 Transfer: P2G

After the addition of the surface and balance particles, we transfer mass and momentum to the grid in the standard APIC [3] way using their conservatively remapped mass and velocity state

$$\begin{aligned}
m_{\mathbf{i}}^n &= \sum_p \tilde{m}_p\left(N_{\mathbf{i}}(\mathbf{x}_p^n) + \sum_{r\in\Pi_p} N_{\mathbf{i}}(\mathbf{s}_r^n) + N_{\mathbf{i}}(\mathbf{b}_r^n)\right),\\
m_{\mathbf{i}}^n\mathbf{v}_{\mathbf{i}}^n &= \sum_p \tilde{m}_p N_{\mathbf{i}}(\mathbf{x}_p^n)\left(\mathbf{v}_p^n + \mathbf{A}_p^n(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n)\right)\\
&\quad + \sum_p \tilde{m}_p \sum_{r\in\Pi_p} N_{\mathbf{i}}(\mathbf{s}_r^n)\left(\mathbf{v}_p^n + \mathbf{A}_p^n(\mathbf{x}_{\mathbf{i}} - \mathbf{s}_r^n)\right)\\
&\quad + \sum_p \tilde{m}_p \sum_{r\in\Pi_p} N_{\mathbf{i}}(\mathbf{b}_r^n)\left(\mathbf{v}_p^n + \mathbf{A}_p^n(\mathbf{x}_{\mathbf{i}} - \mathbf{b}_r^n)\right).
\end{aligned}$$

Here $\mathbf{N}_{\mathbf{i}}(\mathbf{x}) = \mathbf{N}(\mathbf{x} - \mathbf{x}_{\mathbf{i}})$ are quadratic B-splines defined over the uniform grid with $\mathbf{x}_{\mathbf{i}}$ living at cell centers [71]. Note that for interior MPM particles far enough from the boundary that $\Pi_p = \varnothing$. This reduces to the standard APIC [3] splat since $\tilde{m}_p = m_p^n$.

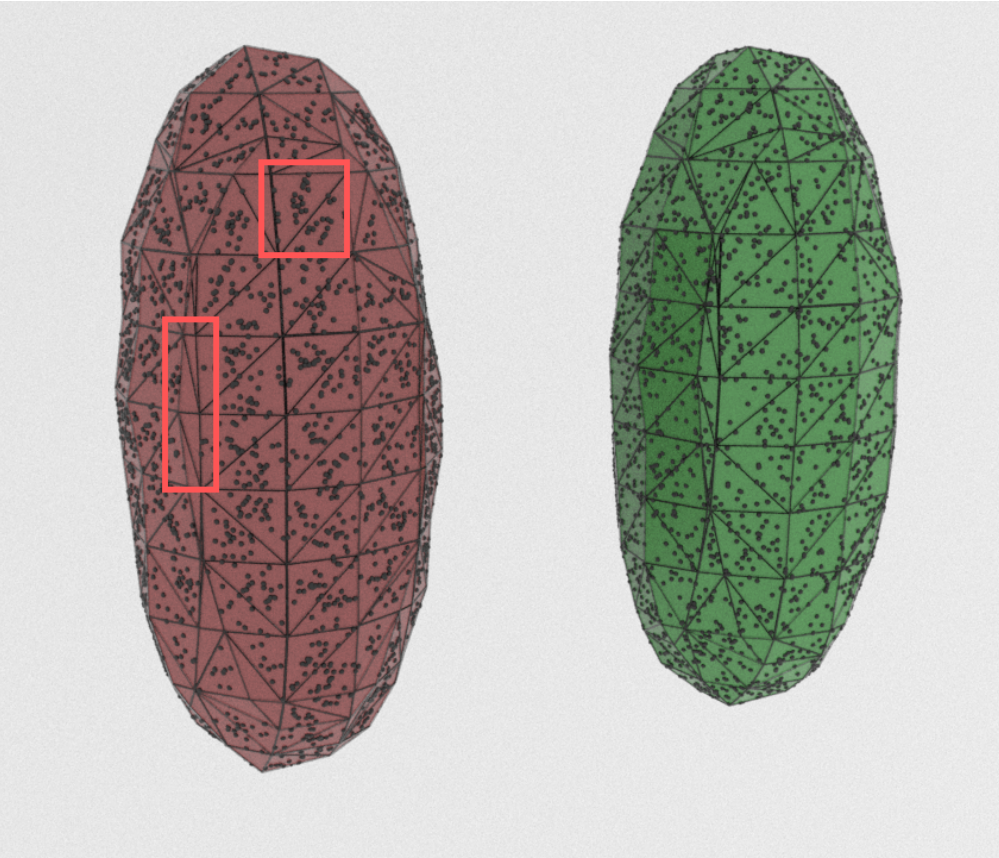


Figure 5.4: **Isocontour and Sampled Boundary Particles**. *(Left)* Low-quality triangles are undersampled and how sample points often clump near triangle centers. *(Right)* The present method, which does not suffer from similar issues.

5.4.5 Grid Momentum Update

We discretize the governing equations in the standard MPM manner by using the particles as quadrature points in the variational forms. The interior MPM particles $\mathbf{x}_p^n$ are used for volume integrals and the surface particles $\mathbf{s}_r^n$ are used for surface integrals. The grid momentum update is :

$$\begin{aligned} m_{\mathbf{i}}^n \tfrac{\hat{\mathbf{v}}_{\mathbf{i}}^{n+1} - \mathbf{v}_{\mathbf{i}}^n}{\Delta t} &= \mathbf{f}_{\mathbf{i}}(\mathbf{x} + \Delta t \hat{\mathbf{q}}) + m_{\mathbf{i}}^n \mathbf{g}, \\ \mathbf{f}_{\mathbf{i}}(\hat{\mathbf{x}}) &= -\tfrac{\partial e}{\partial \hat{\mathbf{x}}_{\mathbf{i}}}(\hat{\mathbf{x}}) - \mu^l \textstyle\sum_p \boldsymbol{\epsilon}^v(\hat{\mathbf{x}}; \mathbf{x}_p^n) \left(\tfrac{\partial N_{\mathbf{i}}}{\partial \mathbf{x}}(\mathbf{x}_p^n)\right)^T V_p^n \end{aligned} \tag{5.21}$$

where $\mathbf{f}_{\mathbf{i}}$ is the force on grid node $\mathbf{i}$ from potential energy and viscosity, $\epsilon^v(\hat{\mathbf{x}}; \mathbf{x}_p^n) =$

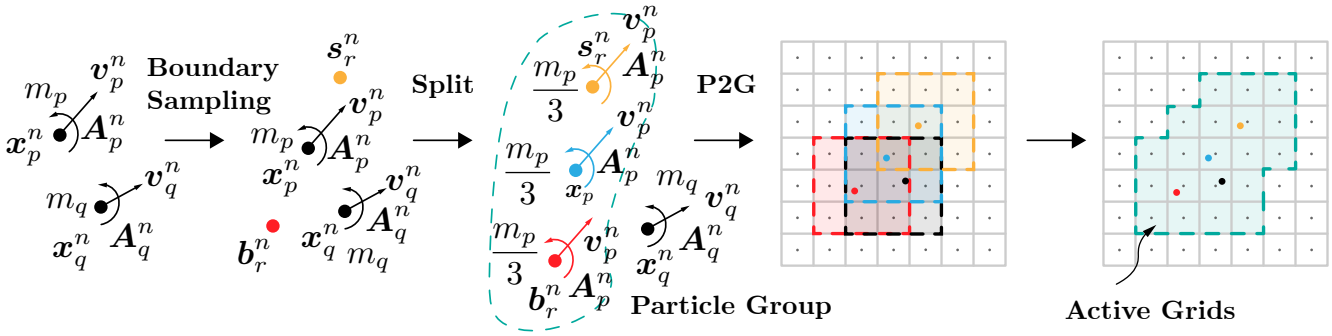


Figure 5.5: **Splitting**. After surface particles (yellow) are created, the mass and momentum of the interior MPM particles (blue) that are closest to the surface particles are immediately distributed. Particles in each particle group are assigned equal mass. MPM particles (black) that are not paired with any surface particles remain intact for the splitting process. Surface particles (yellow) and balance particles (red) are assigned the same linear velocity and affine velocity as the original particle (blue).

$\frac{1}{2}\left(\sum_{\mathbf{j}} \hat{\mathbf{x}}_{\mathbf{j}} \frac{\partial N_{\mathbf{j}}}{\partial \mathbf{x}}(\mathbf{x}_p^n) + \left(\hat{\mathbf{x}}_{\mathbf{j}} \frac{\partial N_{\mathbf{j}}}{\partial \mathbf{x}}(\mathbf{x}_p^n)\right)^T\right)$ is the strain rate at $\mathbf{x}_p^n$, $\mathbf{g}$ is gravity, and $\hat{\mathbf{q}}$ is either 0 (for explicit time integration) or $\hat{\mathbf{v}}^{n+1}$ (for backward Euler time integration). $\mathbf{x}$ represents the vector of all unmoved grid node positions $\mathbf{x}_{\mathbf{i}}$. We use $e(\mathbf{y})$ to denote the discrete potential energy Ψ where MPM and surface particles are used as quadrature points:

$$e(\mathbf{y}) = \sum_p \left(\psi^h(\mathbf{F}_p(\hat{\mathbf{y}})) + \frac{\lambda^l}{2}(J_p(\hat{\mathbf{y}}) - 1)^2\right) V_p^0 + \sum_r k^\sigma(\mathbf{s}_r^n)|\hat{J}_r(\hat{\mathbf{y}})\hat{\mathbf{F}}_r^{-T}(\hat{\mathbf{y}})d\mathbf{A}_r^n|,$$

where, as in [71], $\mathbf{F}_p(\hat{\mathbf{y}}) = \sum_{\mathbf{i}} \mathbf{y}_{\mathbf{i}} \frac{\partial N_{\mathbf{i}}}{\partial \mathbf{x}}(\mathbf{x}_p^n)\mathbf{F}_p^n$ and as in [67], $J_p(\hat{\mathbf{y}}) = \left(1 - d + y_\alpha \frac{\partial N_{\mathbf{i}}}{\partial x_\alpha}(\mathbf{x}_p^n)\right) J_p^n$ and $\hat{\mathbf{F}}_p(\mathbf{y}) = \sum_{\mathbf{i}} \mathbf{y}_{\mathbf{i}} \frac{\partial N_{\mathbf{i}}}{\partial \mathbf{x}}(\mathbf{x}_p^n)$. With these conventions, the α component of the energy-based

force on grid node $\mathbf{i}$ is of the form

$$\begin{aligned}-\frac{\partial e}{\partial x_{\mathbf{i}\alpha}}(\mathbf{y}) = &-\sum_p \frac{\partial \psi^h}{\partial F_{\alpha\delta}}(\mathbf{F}_p(\hat{\mathbf{y}}))F^n_{p\gamma\delta}\frac{\partial N_{\mathbf{i}}}{\partial x_\gamma}(\mathbf{x}^n_p)V^0_p \\ &-\sum_p \lambda^l(J_p(\mathbf{y})-1)\frac{\partial N_{\mathbf{i}}}{\partial x_\alpha}(\mathbf{x}^n_p)J^n_p V^0_p \\ &-\sum_r k^\sigma(\mathbf{s}^n_r)\frac{\partial|\det(\hat{\mathbf{F}}_r)\hat{\mathbf{F}}_r^{-T}d\mathbf{A}^n_r|}{\partial \hat{F}_{\alpha\delta}}(\hat{\mathbf{F}}_r(\hat{\mathbf{y}}))\frac{\partial N_{\mathbf{i}}}{\partial x_\delta}(\mathbf{x}^n_p).\end{aligned} \tag{5.22}$$

We note that the viscous contribution to the force in Equation (5.21) is the same as in [72]. We would expect V^{n+1}_p in this term when deriving from Equation (5.9), however we approximate it as V^n_p. This is advantageous since it makes the term linear; and since $\hat{J}_p(\hat{\mathbf{x}}) \approx 1$ from the liquid pressure and hyperelastic stress, it is not a poor approximation.

5.5 Transfer: G2P

Once grid momentum and temperature have been updated, we transfer velocity and temperature back to the particles. For interior MPM particles with no associated surface or balance particles ($\Pi_p = \varnothing$), we transfer velocity, affine velocity and temperature from the grid to particles in the standard APIC [3] way:

$$\mathbf{v}^{n+1}_p = \sum_{\mathbf{i}} N_{\mathbf{i}}(\mathbf{x}^n_p)\hat{\mathbf{v}}^{n+1}_{\mathbf{i}}, \quad \mathbf{A}^{n+1}_p = \frac{4}{\Delta x^2}\sum_{\mathbf{i}} N_{\mathbf{i}}(\mathbf{x}^n_p)\hat{\mathbf{v}}^{n+1}_{\mathbf{i}}(\mathbf{x}_{\mathbf{i}} - \mathbf{x}^n_p)^T.$$

For interior MPM particles that were split with a collection of surface and balance particles ($\Pi_p \neq \varnothing$), more care must be taken since surface and balance particles will be deleted at the end of the time step. First, the particle is reassigned its initial mass m_p. Then we compute the portion of the grid momentum associated with each surface and balance particle associated with p as

$$\mathbf{p}^s_{\mathbf{i}r} = \tilde{m}_p N_{\mathbf{i}}(\mathbf{s}^n_r)\hat{\mathbf{v}}^{n+1}_{\mathbf{i}}, \ \mathbf{p}^b_{\mathbf{i}r} = \tilde{m}_p N_{\mathbf{i}}(\mathbf{b}^n_r)\hat{\mathbf{v}}^{n+1}_{\mathbf{i}}, \ r \in \Pi_p.$$

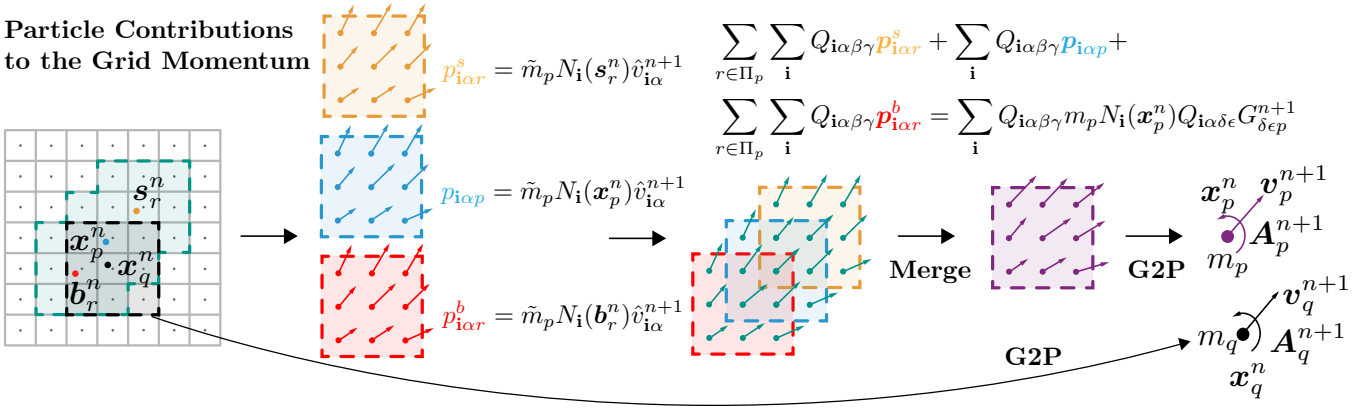


Figure 5.6: **G2P Merging** The merging process is a modified version of G2P. For the particles that are not associated with surface particles (black), a regular G2P is performed. Among each particle group, we calculate each particle's contribution to the grid momentum and the generalized affine moments of their summed momenta about their center of mass. Then, we restore the mass of the original particle associated with the group before the split and compute its generalized affine inertia tensor from its grid mass distribution. Using the affine inertia tensor of the original particle, we compute the generalized velocity of the particle after the merging from the generalized moments of the group.

We then sum this with the split particle's momentum to define the merged particle's momentum

$$\mathbf{p}_{\mathbf{i}p} = \tilde{m}_p N_{\mathbf{i}}(\mathbf{x}^n_p)\hat{\mathbf{v}}^{n+1}_{\mathbf{i}} + \sum_{r\in\Pi_p} \mathbf{p}^s_{\mathbf{i}r} + \mathbf{p}^b_{\mathbf{i}r}.$$

Note that the $\mathbf{p}_{\mathbf{i}p}$ may be nonzero for more grid nodes than the particle would normally splat to (see Figure 5.6). We define the particle velocity from the total momentum by dividing by the mass $\mathbf{v}^{n+1}_p = \frac{1}{m_p}\sum_{\mathbf{i}} \mathbf{p}_{\mathbf{i}p}$. To define the affine particle velocity, we use a generalization of [4] and first compute the generalized affine moments $t_{p\beta\gamma} = \sum_{\mathbf{i}} Q_{\mathbf{i}\alpha\beta\gamma} p_{\mathbf{i}p\alpha}$ of the momentum distribution $p_{\mathbf{i}p\alpha}$ where $Q_{\mathbf{i}\alpha\beta\gamma} = r_{\mathbf{i}p\gamma}\delta_{\alpha\beta}$ is the α component of the $\beta\gamma$ linear mode at grid node $\mathbf{i}$. Here $\mathbf{r}_{\mathbf{i}p} = \mathbf{x}_{\mathbf{i}} - \mathbf{x}^n_p$ is the displacement from the center of mass of the distribution to the grid node $\mathbf{x}_{\mathbf{i}}$. We note that these moments are the generalizations of angular momentum

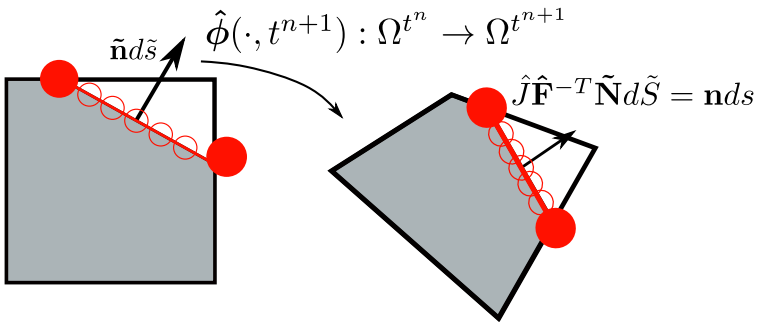


Figure 5.7: **Normals**. *(Left)* The isocontour face normal with each sampled points being provided a surface area-weighted normal. *(Right)* The isocontour face normals and areas are updated with the mapping between t^n and t^{n+1}.

to affine motion, as was observed in [3], however in our case we compute the moments from a potentially wider distribution of momenta $p_{\mathbf{i}p\alpha}$. Lastly, to conserve angular momenta (see [73] for details), we define the affine velocity by inverting the generalized affine inertia tensor $\sum_{\mathbf{i}} Q_{\mathbf{i}\alpha\gamma\delta} m_p N_{\mathbf{i}}(\mathbf{x}_p^n) Q_{\mathbf{i}\alpha\epsilon\tau}$ of the point $\mathbf{x}_p^n$ using its merged mass distribution $m_p N_{\mathbf{i}}(\mathbf{x}_p^n)$. However, as noted in [3], the generalized inertia tensor $\frac{m_p \Delta x^2}{4}\mathbf{I}$ is constant diagonal when using quadratic B-splines for $N_{\mathbf{i}}(\mathbf{x}_p^n)$ and therefore the final affine velocity is $\mathbf{A}_p^{n+1} = \frac{4}{m_p \Delta x^2}\mathbf{t}_p$.s

5.6 Surface Tension Boundary Sampling

The approach for seeding the surface tension quadrature particles $\mathbf{x}_q^n$ with area weighted normals $d\mathbf{A}_q$ on $\partial\Omega^{t^n}$ is described. As is common in particle-based methods (e.g. [74]), a spherical level set is defined around each fluid particle and the union of the individual level sets form an implicit representation φ of Ω^{t^n}. Then $\partial\Omega^{t^n}$ is defined as the zero isocontour of φ. However, the isocontour may be relatively distant from the fluid particles due to the level set radii of the particles (typically set as approximately $.4\Delta x$). A uniform shift φ by a multiple of Δx before computing the isocontour is used(see Figure 5.2). Additionally, performing one iteration of Laplace smoothing on φ resulted in smoother isocontours which are more desirable when applying surface tension forces.

After creating the isocontours, $\mathbf{x}_q^n$ and $d\mathbf{A}_q$ are sampled on each face (see Figure 5.2).

In two dimensions, sampling of equally spaced points along each face is used across the entire isocontour. In three dimensions, multiple random barycentric coordinates on the sampled face are used. In both two and three dimensions, the surface area-weighted normal $d\mathbf{A}_q$ are assigned such that the surface area of an isocontour face is evenly distributed among sample points (see Figure 5.7).

5.7 Implicit Formulation

In the implicit case, we solve equation 5.16 with $\hat{\mathbf{q}} = \hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$,

$$m_{\mathbf{i}}^n \frac{\hat{v}_{\mathbf{i}\alpha}^{n+1} - v_{\mathbf{i}\alpha}^n}{\Delta t} = f_{\mathbf{i}\alpha}(\mathbf{x} + \Delta t \hat{\mathbf{v}}^{n+1}) + m_{\mathbf{i}}^n g_\alpha, \tag{5.23}$$

for $\hat{\mathbf{v}}^{n+1}$ using Newton's method. We solve the resulting system using conjugate gradient. This requires the derivative of the force

$$\frac{\partial f_{\mathbf{i}\alpha}}{\partial \hat{x}_{\mathbf{j}\beta}}(\hat{\mathbf{x}}) = -\sum_p \frac{\partial^2 \hat{\Psi}^p}{\partial J^2} J_p^n \frac{\partial N_{\mathbf{i}}}{\partial x_\alpha} J_p^n \frac{\partial N_{\mathbf{j}}}{\partial x_\beta} V_p^n - \sum_q \frac{\partial^2 \hat{\Psi}^s}{\partial \hat{F}_{\beta\delta} \partial \hat{F}_{\alpha\gamma}} \frac{\partial N_{\mathbf{i}}}{\partial x_\gamma} \frac{\partial N_{\mathbf{j}}}{\partial x_\delta}. \tag{5.24}$$

The Hessian $\partial^2\hat{\Psi}^2/\partial\hat{\mathbf{F}}^2$ of the surface tension energy density is indefinite in three dimensions. However, we can analytically determine its eigenstructure as

Eigenvalue	Eigenvectors
$k^\sigma\lvert d\mathbf{A}\rvert$	$\frac{1}{\sqrt{2}}\mathbf{b}_1 \otimes \mathbf{b}_2 + \frac{1}{\sqrt{2}}\left(\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \times \mathbf{b}_1\right) \otimes \left(\frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert} \times \mathbf{b}_2\right)$ $\frac{1}{\sqrt{2}}\mathbf{b}_1 \otimes \left(\frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert} \times \mathbf{b}_2\right) - \frac{1}{\sqrt{2}}\left(\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \times \mathbf{b}_1\right) \otimes \mathbf{b}_2$
$-k^\sigma\lvert d\mathbf{A}\rvert$	$\frac{1}{\sqrt{2}}\mathbf{b}_1 \otimes \mathbf{b}_2 - \frac{1}{\sqrt{2}}\left(\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \times \mathbf{b}_1\right) \otimes \left(\frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert} \times \mathbf{b}_2\right)$ $\frac{1}{\sqrt{2}}\mathbf{b}_1 \otimes \left(\frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert} \times \mathbf{b}_2\right) + \frac{1}{\sqrt{2}}\left(\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \times \mathbf{b}_1\right) \otimes \mathbf{b}_2$
$k^\sigma \frac{\lvert d\mathbf{A}\rvert^2}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert}\lvert\hat{\mathbf{F}}\mathbf{w}_1\rvert^2$	$\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \otimes \mathbf{w}_0$
$k^\sigma \frac{\lvert d\mathbf{A}\rvert^2}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert}\lvert\hat{\mathbf{F}}\mathbf{w}_0\rvert^2$	$\frac{\hat{J}\hat{\mathbf{F}}^{-T}}{\lvert\hat{J}\hat{\mathbf{F}}^{-T}\rvert} \otimes \mathbf{w}_1$
0	$\mathbf{u}_0 \otimes \frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert}$ $\mathbf{u}_1 \otimes \frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert}$ $\mathbf{u}_2 \otimes \frac{d\mathbf{A}}{\lvert d\mathbf{A}\rvert}$

(5.25)

where $\mathbf{b}_1$ and $\mathbf{b}_2$ are any unit vectors orthogonal to $d\mathbf{A}$ and $\hat{J}\hat{\mathbf{F}}^{-T}$ respectively, $\mathbf{w}_0$ and $\mathbf{w}_1$ are any orthonormal vectors orthogonal to $d\mathbf{A}$ satisfying $\hat{\mathbf{F}}\mathbf{w}_0 \cdot \hat{\mathbf{F}}\mathbf{w}_1 = 0$, and $\mathbf{u}_0$, $\mathbf{u}_1$, and $\mathbf{u}_2$ are any orthonormal basis for $\mathbb{R}^3$. We refer readers to the supplementary material [75] for a proof. Note that there is one negative eigenvalue with multiplicity two and a zero eigenvalue with multiplicity three. We perform a definiteness fix as in [76, 77, 78] by clamping these negative eigenvalues to 0. The corresponding term in the Jacobian matrix for Newton's method will then be positive semi-definite.

5.8 Results

5.8.1 Massless and Mass Resampling Comparasion

The massless particles can poorly integrate the traction force when the isocontour of the massless particles is outside the particles' background grid. This error in the traction will break the conservation of linear momentum. A comparison is made in 5.8 to show how the

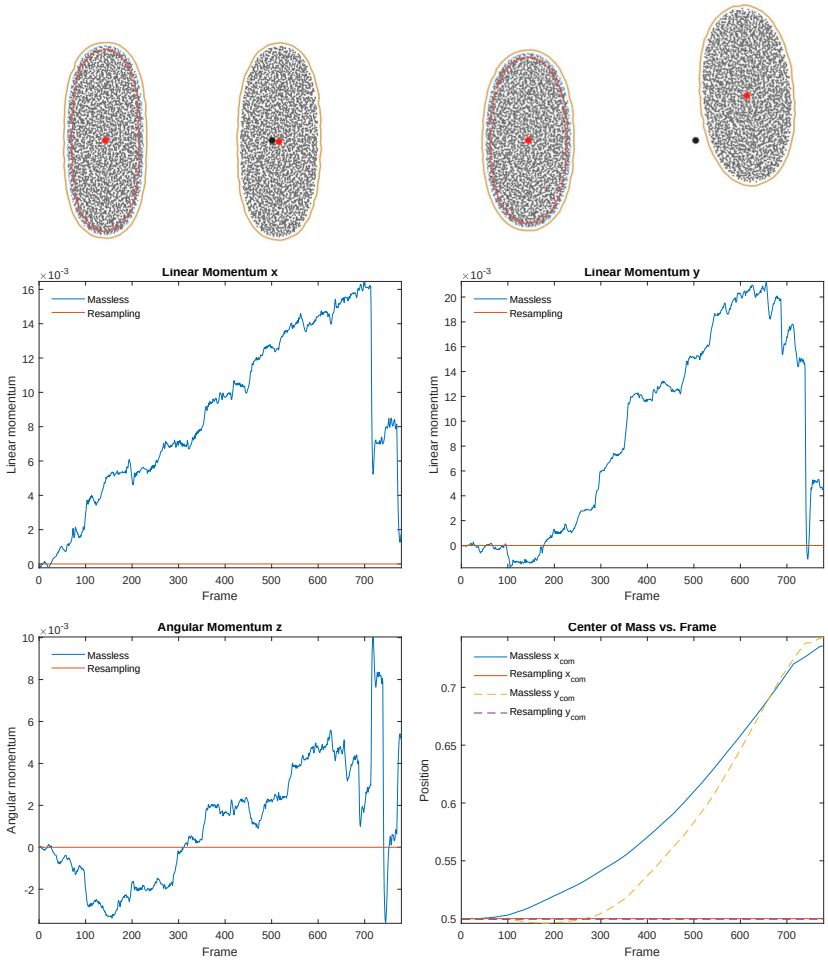


Figure 5.8: **Conservation of Momemtum**. Comparison for resampling and massless approaches with the isocontour outside the particles.

errors in the surface tension forces break the simulation symmetry as well as the total linear and angular momentum. To avoid this behavior with the massless approach the isocontour must be brought inside the particles. However, sharp features of the flow are smoothed. Here resampling is used to guarantee the traction forces are integrated properly and the total conservation of linear and angular momentum is maintained.

5.8.2 Dropping Spheres

We simulate several inviscid spherical droplets with radii of 0.15m free fall and impact a dry surface. In Figure 5.9, droplets with different surface tension coefficients drop from a height of 2.5m. The size of the simulation domain is 1m $\times$ 3m $\times$ 1m. With different surface tension coefficients k^σ, the droplets display distinct behaviors upon impact, as shown in Figure 5.10. We also capture the partial rebound and the rebound behaviors of the droplet after the impact. The middle and bottom rows of Figure 5.10 show the footage of a droplet with $k^\sigma = 15\text{N/m}$ dropped from a height of 3.5m (in 1m $\times$ 4m $\times$ 1m domain) and a droplet with $k^\sigma = 5\text{N/m}$ dropped from a height of 2.5m (in 1m $\times$ 3m $\times$ 1m domain), respectively. Our results qualitatively match the experiment outcomes from [79].

Figure 5.9: **Spherical Drops**. Spherical droplets with different surface tension coefficients free fall from the same height. In the top figure, from left to right, the surface tension coefficients are $k^\sigma = 20, 5, 1, 0.1, 0.05\text{N/m}$.

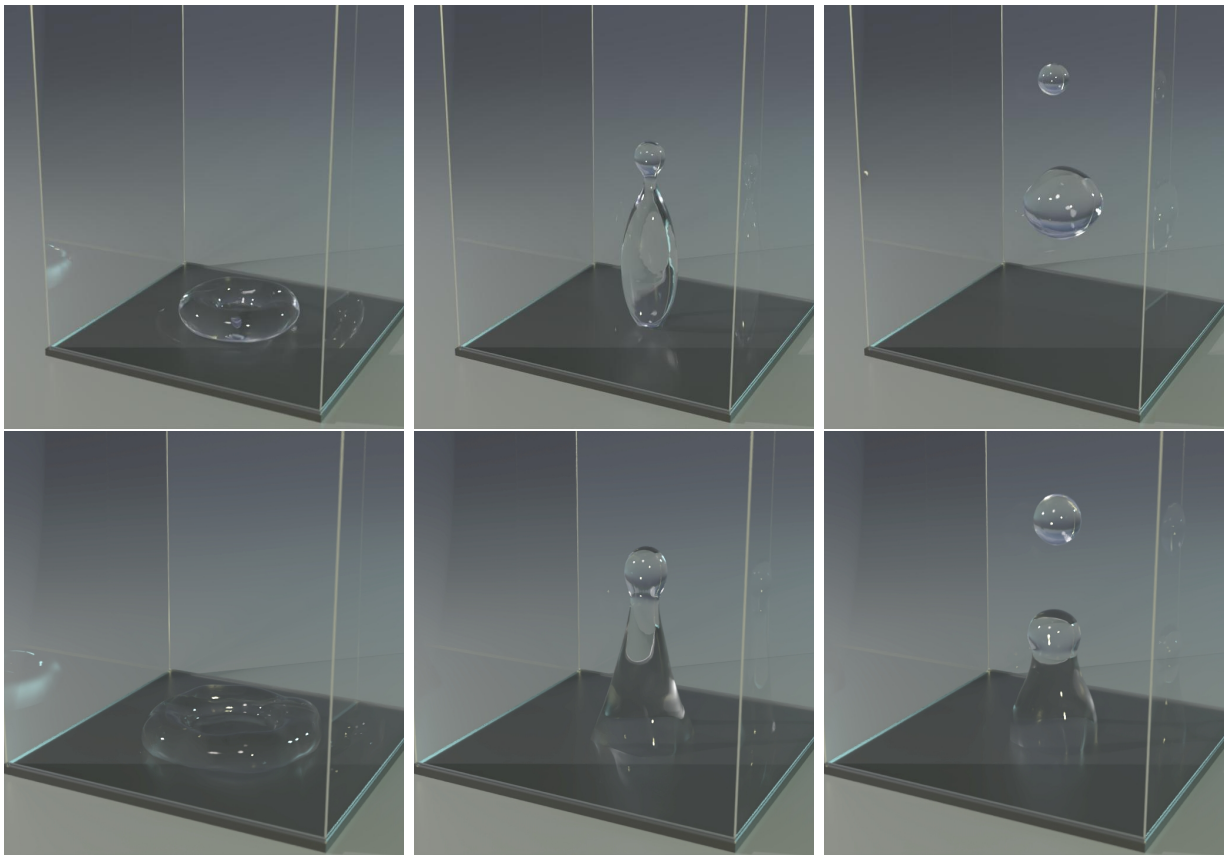


Figure 5.10: **Spherical Drops Time Lapse**. *(Top Row)* full rebound of the droplet (initial height: 3.5m and $k^\sigma = 15\text{N/m}$). *(Bottom Row)* partial rebound of the droplet (initial height: 2.5m and $k^\sigma = 5\text{N/m}$).

5.8.3 Contact Angles

The contact angle between a liquid, a solid boundary, and the ambient air is governed by the Young equation [80]. This expression relates the resting angle θ (measured through the liquid) of a liquid in contact with a solid surface to the surface tension coefficients between the liquid, solid, and air phases:

$$k^\sigma_{SG} = k^\sigma_{SL} + k^\sigma_{LG}\cos(\theta). \tag{5.26}$$

The surface tension coefficients are between the solid and gas phases, solid and liquid phases, and liquid and gas phases, respectively. As in [81], we assume k^σ_{SG} is negligible. Under

this assumption, the solid-liquid contact angle is determined by the surface tension ratio $-k^{\sigma}_{SL}/k^{\sigma}_{LG}$.

Figure 5.11 shows that our method enables simulation of various contact angles, emulating various degrees of hydrophobic or hydrophilic behavior as a droplet settles on a surface. We adjust the contact angles by assigning one surface tension coefficient, k^{σ}_{LG}, to the surface particles on the liquid-gas interface, and another one, k^{σ}_{SL}, to those on the solid-liquid interface.

A droplet of radius 0.1m is placed right above the ground in a 0.5m $\times$ 0.5m $\times$ 0.5m domain. Each droplet is discretized using 230k interior particles and 250k surface particles. The surface tension k^{σ}_{LG} is set to 2N/m, and we approximate k^{σ}_{SL} based on the desired contact angle θ. A dynamic viscosity of 0.075Pa $\cdot$ s is used to stabilize the simulations.

Figure 5.12 shows an example of several liquid drops with different k^{σ} ratios falling on ramps of 5.5° angle. The length of the ramp is 3m, and the domain size is 3m $\times$ 0.5m $\times$ 1m. Coulomb friction with a friction coefficient of 0.2 was used for the ramp surface, and the drop has no viscosity. When there is a larger difference between solid-liquid and liquid-air surface tension coefficients (i.e., a smaller k^{σ} ratio), the liquid tends to drag more on the surface and undergo more separation and sticking. The leftmost example, with a k^{σ} ratio of 1.0, exhibits hydrophobic behavior.

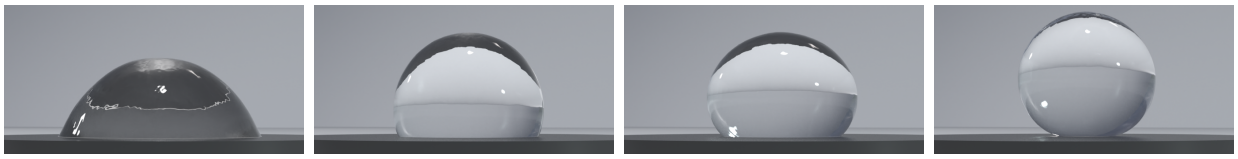


Figure 5.11: **Stationary Drops**. As our droplets settle, we are able to obtain contact angles of approximately 45, 90, 135 and 180 degrees, using a $k^{\sigma}_{SL}/k^{\sigma}_{LG}$ ratio of $-\sqrt{2}/2$, 0, $\sqrt{2}/2$ and 1, respectively.

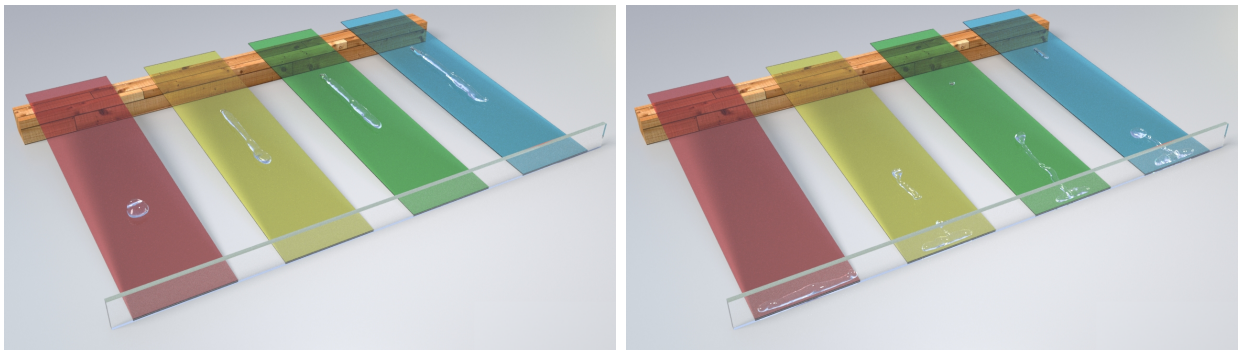


Figure 5.12: **Drops Slidding in Ramp**. Liquid drops fall on a ramp with varying ratios between the solid-liquid and liquid-air surface tension coefficients. From left to right: ratios of 1.0, 0.6, 0.3, 0.05. *(Top)* Frame 60. *(Bottom)* Frame 100.

5.8.4 Two Way Coupling

For two-way coupling between fluids and elastic meshes, we leverage the Lagrangian force framework of [3]. In this figure, a hyper-elastic mesh is coupled with a surface tension simulation. This figure shows a direct coupling between MPM and the mesh.

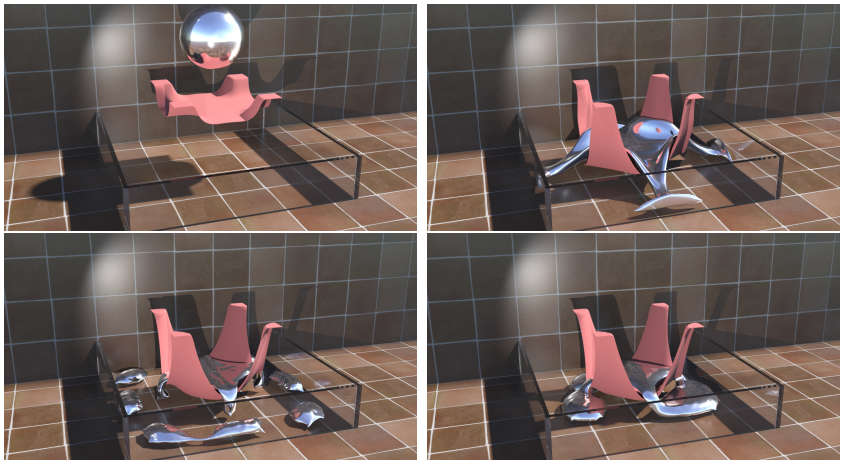


Figure 5.13: **Two Way Coupling** of high surface tension and high internal energy with an hyper-elastic mesh

5.8.5 Performance Considerations

The table shows average per-timestep runtime details for several of our examples. For this table, all experiments were run on a workstation equipped with 128GB RAM and with dual Intel® Xeon® E5-2687W v4 CPUs at 3.00Ghz.

Example	# Cells	# Int. Part.	# Surf. Part.	Sampling	Merging	Part.→Grid	Grid→Part.	Linear Solve	Time Step
Droplet Impact ($k^\sigma = 5$)	2M	794K	100K	2224	20	95	39	1422	10065
Droplets on Ramps ($k^\sigma_{SL}/k^\sigma_{LG} = 0.05$)	1.5M	70K	100K	258	6	28	9	199	1434
Contact Angles ($k^\sigma_{SL}/k^\sigma_{LG} = 0$)	256K	230K	250K	492	17	73	38	647	4286

5.9 Discussion and Limitations

Instead of being fully incompressible, the formulation is based on the compressible Euler equations with a penalty on compression. An incompressible formulation would require the solution of a nonlinear KKT system, which would take more time to solve but may still be preferable. The current Newton system for the implicit formulation is solved using a conjugate gradient with no preconditioner. This does not present a hurdle for solving the system, as in practice we require few CG iterations per Newton step. Nonetheless, the application of a good preconditioner could improve the convergence of the CG iteration and reduce the solve time for each step. Both our MPM discretization and our level-set-based boundary sampling technique are conceptually simple and hence relatively easy to implement in contrast to most front tracking or unstructured discretizations which require dynamic remeshing. However, such approaches are more capable of maintaining sharp interfaces and are likely more accurate results. Our method allows for the simulation of surface tension energies with spatial gradients, including those driven by variation in temperature. Our MPM approach to the problem resolves many interesting characteristic phenomena associated with these variations. Adding in a mixture model as in [82] would be interesting future work. Lastly, although our approach was designed for MPM, SPH is more commonly used for the simulation of liquids.

However, SPH and MPM have many similarities, as recently shown by the work of Gissler et al. [83], and it would be interesting future work to generalize our approach to SPH.

5.10 Future Works

Coupling with thermal effects would be the next step. The particle properties can have temperature dependence that is needed to model melting and phase changes in the material. The approach supports variation in the material properties such as viscosity, elasticity, and surface tension. Extensions have been currently made, but an extended study on the characteristics of the method and applications to engineering problems remains to be done.

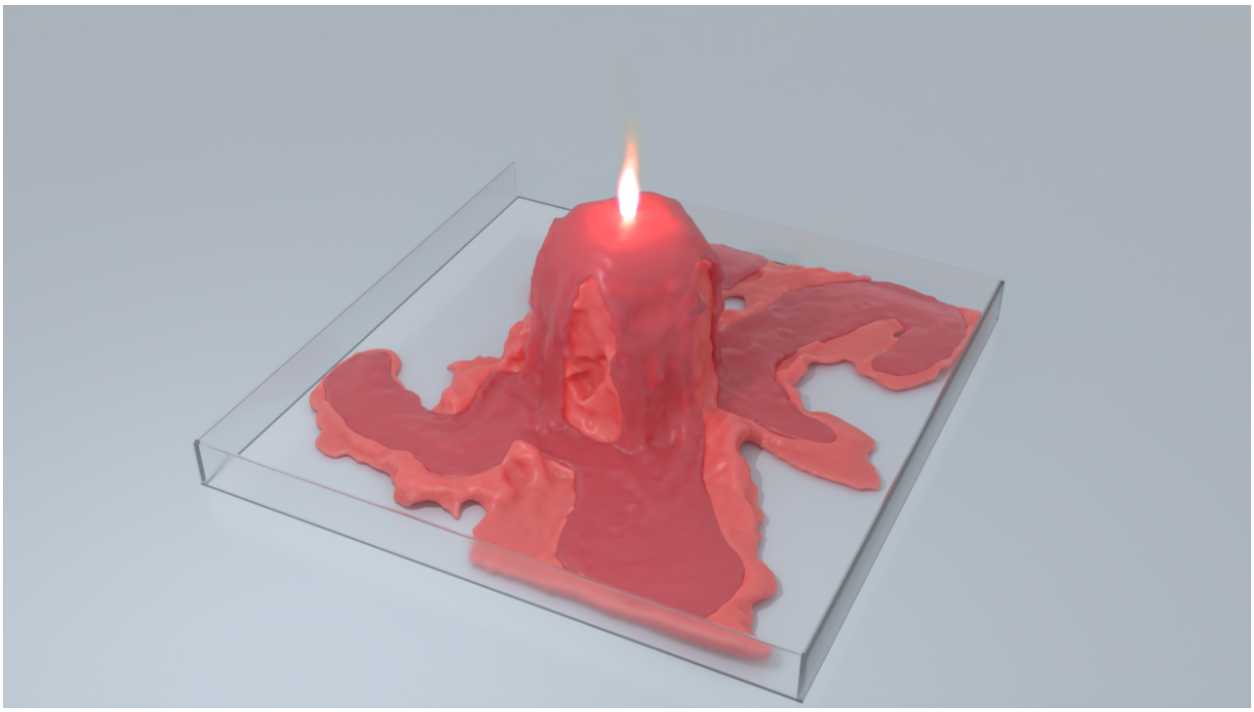


Figure 5.14: **Candle** is used to show the applications of the method in modeling temperature phase change problems with surface tension.

CHAPTER 6

A Hybrid Lagrangian/Eulerian Collocated Advection and Projection Method for Fluid Simulation

Hybrid methods that use MPM can take full advantage of both representations to improve the performance and computational cost. The incompressible assumption to approximate fluids is accurate when modeling fluids that are similar to water. The Chorin [84] splitting of advection and pressure projection has been a well-established technique to incorporate the incompressibility. This method is typically used alongside regular grids of Marker-And-Cell (MAC) [85] type with pressure and velocity components staggered at cell centers and faces respectively. Furthermore, advection is most often discretized using semi-Lagrangian techniques originally developed in the atmospheric sciences [86, 87]. Although well-established, these techniques are not without their drawbacks. Grid staggering can complicate many algorithms related to incompressible flow. E.g. Particle-In-Cell (PIC) [88] techniques like FLIP [89, 90], Affine/Polynomial Particle-In-Cell (APIC/PolyPIC) [3, 4] and the Material Point Method (MPM) [65, 91] which information must be transferred to and from each individual grid.

Another limitation of the MAC grid arises with free-surface water simulation. Where the staggering prevents many velocity components near the fluid free surface from receiving a correction during projection (see e.g. [92]). However, MAC grids are useful because the staggering prevents pressure null modes while allowing for accurate second order central differencing in discrete grad/div operators.

Alternatives include mixed Finite Element Method (FEM) techniques that use collocated velocities [93] without suffering from pressure mode instabilities. One example are Taylor-Hood elements [94] which use collocated multi-quadratic velocity interpolation and multilinear pressure interpolation to enforce incompressibility. Recently, B-spline interpolation [95] has been used with Taylor-Hood [96]. Expanding on this approach a method using collocated multi-quadratic B-spline interpolation for velocities was developed. The C^1 interpolation from the B-splines is essential for stability [97]. This B-spline property is used to develop BSLQB a novel Backward Semi-Lagrangian (BSL) [87] technique that achieves second order accuracy in space and time. The BSL method for quadratic B-splines dramatically reduces numerical dissipation with only a small modification to the widely-adopted explicit semi-Lagrangian formulations typically used in graphics applications. BSL techniques utilize the implicit form of semi-Lagrangian advection. Semi-Lagrangian techniques for velocity advection utilize the implicit relation associated with solution of Burgers' equation.

$$\mathbf{u}(\mathbf{x},t) = \mathbf{u}(\mathbf{x} - (t-s)\mathbf{u}(\mathbf{x},t), s) \iff \frac{D\mathbf{u}}{Dt} = \frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{u} = \mathbf{0} \tag{6.1}$$

for $s \leq t$ [98]. Traditionally, graphics applications have preferred the explicit variant of semi-Lagragian advection whereby grid velocities are updated through the expression

$$\mathbf{u}_\mathbf{i}^{n+1} = \mathbf{u}(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{u}_\mathbf{i}^n, t^n) \tag{6.2}$$

where $\mathbf{x_i}$ is the location of grid node $\mathbf{i}$, $\mathbf{u}_\mathbf{i}^n, \mathbf{u}_\mathbf{i}^{n+1}$ are velocities at the node at times t^n and t^{n+1} respectively and interpolation over the velocity grid is used to estimate $\mathbf{u}(\mathbf{x_i} - \Delta t \mathbf{u}_\mathbf{i}^n, t^n)$ at non-grid node locations [99, 86]. In contrast, BSL techniques leverage Equation (6.1) directly

$$\mathbf{u}_\mathbf{i}^{n+1} = \mathbf{u}(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{u}_\mathbf{i}^{n+1}, t^n) \tag{6.3}$$

Which requires the solution of an implicit equation for $\mathbf{u}_\mathbf{i}^{n+1}$ [87]. Since our grid interpolation is C^1 solving the system requires only a few steps of Newton's method. While this is more expensive than the explicit semi-Lagrangian formulations, each node can still be updated in

parallel since the implicit equations for $\mathbf{u}_\mathbf{i}^{n+1}$ are decoupled in $\mathbf{i}$. The solution of the implicit Equation (6.3), rather than the explicit Equation (6.2) improves the order of convergence from first to second (in space and time). This does not require use of multiple time steps for backward/forward estimations of error, as is commonly done [100, 101, 102, 103, 104]. Furthermore, the method allows for larger-than-CFL time steps and is as stable or more so than explicit semi-Lagrangian formulations.

Lastly, a hybrid particle/BSLQB advection technique that utilizes APIC/PolyPIC [4] in portions of the domain covered by particles and BSLQB in portions without particles was developed. Our formulation naturally leverages the strengths of both approaches. Dense concentrations of particles can be added to regions of the domain where more detail is desired. Also, if particle coverage becomes too sparse because of turbulent flows, BSLQB can be used in the gaps. We demonstrate this technique with smoke simulation and narrow banding of particles near the fluid surface with water simulations as in [105, 106, 107]. In this case, level set advection naturally enabled with our BSLQB formulation is preferred in deeper water regions. The contribution are:

- A novel collocated velocity B-spline mixed FEM method for Chorin [84] splitting discretization of the incompressible Euler equations.
- BSLQB: a novel BSL technique designed for collocated multiquadratic B-spline velocity interpolation that achieves second order accuracy in space and time.
- A hybrid BSLQB/PolyPIC method for narrow band free-surface flow simulations and concentrated-detail smoke simulations.

6.1 Related Work

Advection: Stam [86] first demonstrated the efficacy of semi-Lagrangian techniques for graphics applications and they have since become the standard. Many modifications to Stam [86] have been developed. Fedkiw et al. [108] use vorticity confinement [109] to counterbalance vorticity lost to dissipation and cubic grid interpolation. Kim et al. [100, 101] and Selle et al. [102] combine forward and backward semi-Lagrangian steps to estimate and remove dissipative errors. Constrained Interpolation Profile [110, 111, 112] techniques additionally advect function derivatives to reduce dissipation. Molemaker et al. [113] use the QUICK technique of Leonhard [114] which is essentially upwinding with quadratic interpolation and Adams-Bashforth temporal discretization. Backward Difference Formula techniques are useful because they use an implicit multistep formulation for higher-order semi-Lagrangian advection yet still only require one projection per time step [103, 104].

Semi-Lagrangian techniques interpolate data from a characteristic point. This idea goes back to the Courant-Issaacson-Rees [115] method. However, as noted in [108] semi-Lagrangian advection is very popular in atmospheric science simulation and the variants used in graphics that account for characteristics traveling beyond the local cell in one time step go back to Sawyer [99]. The first BSL approach utilizing Equation (6.3) was done by Robert [87] in which they use fixed point iteration to solve the nonlinear equation. They fit a bicubic function to their data over 4×4 grid patches, then use that function in the fixed point iteration. If the upwind point leaves the grid, they clamp it to the boundary of the 4×4 patch. This clamping will degrade accuracy for larger time steps. In this case, more general interpolation is typically used (see [116, 117] for useful reviews). Pudykiewicz and Staniforth [118] investigate the effects of BSL versus explicit semi-Lagrangian. Specifically, they compare Bates and McDonald [119] (explicit) versus Robert [87] (BSL). They show that keeping all things equal, the choice of Equation (6.2) (explicit) instead of Equation (6.3) (BSL) leads to more dissipation and mass loss. This is consistent with our observations with BSLQB.

Interestingly, multiquadratic B-splines are not often utilized instead Hermite splines, multicubic splines and even Lagrange polynomials are commonly used [116]. Preference for Hermite splines and Lagrange polynomials is likely due to their local nature (they do not require solution of a global system for coefficients) and preference for multicubic splines (over multi-quadratic) is possibly due to the requirement of odd degree for natural splines (odd degree splines behave like low pass filters and tend to be smoother than even degree splines [120, 121]). Cubic splines are considered to be more accurate than Hernite splines and Lagrange interpolation [116, 122]. Interestingly, Riishøjgaard et al. [123] found that cubic spline interpolation gave rise to a noisier solution than cubic Lagrange interpolation with a technique analogous to that of Makar and Karpik [122]. However, they also note that addition of a selective scale diffusion term helps reduce noise associated with cubic splines. Wang and Layton [124] use linear B-splines with BSL but only consider one space dimension which makes Equation (6.3) linear and easily solvable.

Dissipation with explicit semi-Lagrangian advection is so severe that many graphics researchers have resorted to alternative methods to avoid it. Mullen et al. [125] develop energy preserving integration to prevent the need for correcting dissipative behavior. Some authors [126, 127, 128, 129] resolve the flow map characteristics for periods longer than a single time step (as opposed to one step with semi-Lagrangian) to reduce dissipation. Hybrid Lagrangian/Eulerian techniques like PIC (and related approaches) [92, 3, 4, 90] explicitly track motion of particles in the fluid, which is nearly dissipation-free, but can suffer from distortion in particle sampling quality. Vorticity formulations are also typically less dissipative, but can have issues with boundary conditions enforcement [130, 131, 132, 133, 134, 135]. Zehnder et al., Zhang et al. and Mullen et al. [125, 136, 137, 138] have noted that the Chorin projection itself causes dissipation. Zhang et al. [138] reduced artificial dissipation caused by the projection step by estimating lost vorticity and adding it back into the fluid.

Zehnder et al. [136, 137] propose a simple, but very effective modification to the splitting scheme that is similar to midpoint rule integration to reduce the projection error.

Pressure projection: Graphics techniques utilizing pressure projection typically use voxelized MAC grids with boundary conditions enforced at cell centers and faces, however many methods improve this by taking into account sub-cell geometric detail. Enright et al. [139] showed that enforcing the pressure free surface boundary condition at MAC grid edge crossings (rather than at cell centers) dramatically improved the look of water surface waves and ripples. Batty, Bridson and colleagues developed variational weighted finite difference approaches to enforce velocity boundary conditions with MAC grids on edge crossings and improved pressure boundary conditions at the free surface in the case of viscous stress [140, 141, 142]. XFEM [143, 144] and virtual node (VNA) [104] techniques also use cut-cell geometry with variational techniques. Schroeder et al. [104] use cut-cells with MAC grids, but their technique is limited to moderate Reynolds numbers.

There is a vast literature on enforcing incompressibility in the FEM community [93]. Our approach is most similar to the B-spline Taylor-Hood element of Bressan [96]. Adoption of B-spline interpolation in FEM is part of the isogeometric movement [145, 146]. Originally motivated by the desire to streamline the transition from computer-aided design (CAD) to FEM simulation, isogeometric analysis explores the use of CAD-based interpolation (e.g. B-splines and nonuniform rational B-splines (NURBS)) with FEM methodologies. Hughes et al. [145] show that in addition to simplifying the transition from CAD to simulation, the higher regularity and spectral-like properties exhibited by these splines makes them more accurate than traditionally used interpolation. We enforce Dirichlet boundary conditions weakly as in XFEM and VNA approaches [143, 144, 104]. Bazilevs et al. [147] show that weak Dirichlet enforcement with isogeometric analysis can be more accurate than strong enforcement.

Graphics applications are typically concerned with turbulent, high-Reynolds numbers flows.

Interestingly, B-splines have proven effective for these flows by researchers in the Large Eddy Simulation (LES) community [148, 149]. Kravchenko et al. [149] use a variational weighted residuals approach with B-splines for turbulent LES and show that the increased regularity significantly reduces computational costs. Boatela et al. [150] use a similar approach, but apply a collocation technique where the strong form of the div-grad formulation of incompressibility is enforced point wise. They show that their B-spline approach attains optimal order of accuracy of the quadratic flow invariants. Boatela et al. [150] also introduce a notion of sparse approximation to the inverse mass matrix to avoid dense systems of equations in the pressure solve.

6.2 Governing Equations and Operator Splitting

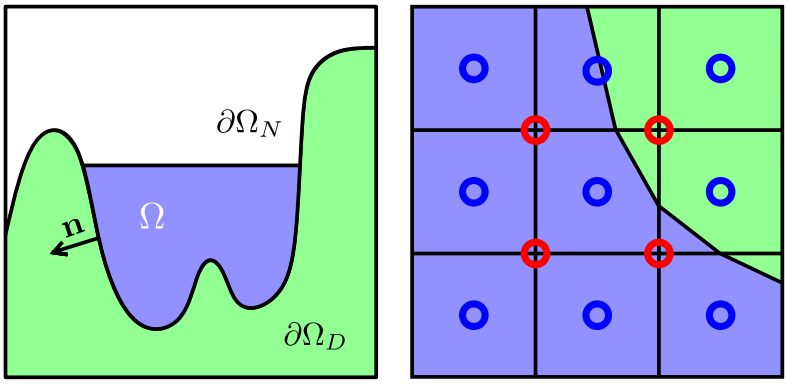


Figure 6.1: **Flow Domain and Grid Left**: we use Ω to denote the fluid domain, with $\partial\Omega_D$ used to indicate the portion of the fluid domain subject to velocity boundary conditions and $\partial\Omega_N$ to indicate the free-surface portion of the boundary with pressure condition $p = 0$. **Right**: We use multiquadratic interpolation for velocity ($\bar{\mathbf{u}}_\mathbf{i}$ at cell centers, blue) and multilinear for pressure ($p_\mathbf{c}$ at nodes, red). The fluid domain is defined with sub-grid-cell accuracy.

.

The incompressible Euler equations are the conservation of linear momentum, conservation of mass which gives the incompressible constraint, the no slip condition and the free surface boundary condition. Ω represents the fluid domain, $\partial\Omega_D$ represents the boundary section of the fluid domain which velocity is prescribed to a (which may vary over the boundary) and $\partial\Omega_N$ is the boundary section where the pressure is zero (see Figure 6.1).

$$\rho\frac{D\mathbf{u}}{Dt} = \rho\left(\frac{\partial\mathbf{u}}{\partial t} + \frac{\partial\mathbf{u}}{\partial\mathbf{x}}\mathbf{u}\right) = -\nabla p + \rho\mathbf{g},\ \mathbf{x}\in\Omega \tag{6.4}$$

$$\nabla\cdot\mathbf{u} = 0,\ \mathbf{x}\in\Omega \tag{6.5}$$

$$\mathbf{u}\cdot\mathbf{n} = a,\ \mathbf{x}\in\partial\Omega_D \tag{6.6}$$

$$p = 0,\ \mathbf{x}\in\partial\Omega_N \tag{6.7}$$

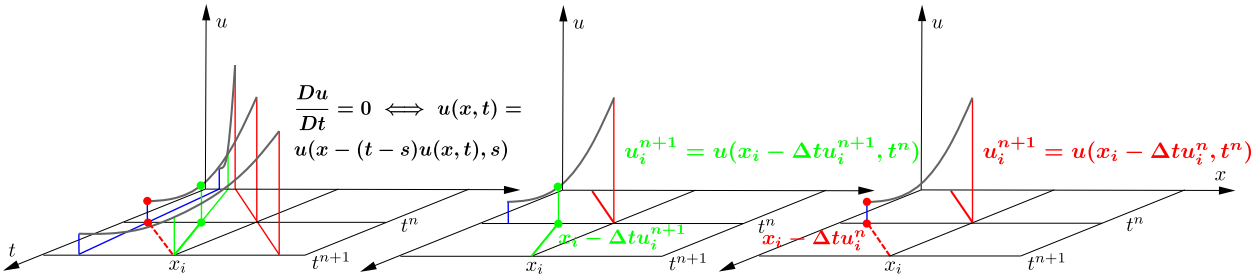


Figure 6.2: **BSL vs. SL**. We illustrate the difference between explicit semi-Lagrangian and BSL in 1D. **Left**: The exact solution of Burgers' equation has straight line characteristics shown in blue, green and red on which velocity (plotted above the plane in gray) is constant. **Center**: BSL (green) uses Newton's method to solve for the exact characteristic going through x_i at time t^{n+1} to determine u_i^{n+1}. **Right**: explicit semi-Lagrangian (red) uses a stale, time t^n approximation of the characteristic which over shoots, resulting in an underestimate of the velocity and energy loss.

In a Chorin [84] operator splitting of the advective and pressure terms, velocity is first updated to an intermediate field $\mathbf{w}$ under the convective $\rho\frac{D\mathbf{u}}{Dt} = \mathbf{0}$, followed by an update from the pressure and gravitational body forcing under $\rho\frac{\partial\mathbf{u}}{\partial t} = -\nabla p + \rho\mathbf{g}$ where the pressure

is determined to enforce $\nabla \cdot \mathbf{u} = 0$. Dividing by the mass density, the convective step is seen to be an update under Burgers' equation6.1. Burgers' equation governs temporally constant Lagrangian velocity (zero Lagrangian acceleration). The characteristic curves for flows of this type are straight lines (since the Lagrangian acceleration is zero), on which the velocity is constant (see Figure 6.2). This gives rise to the implicit relation $\mathbf{u}(\mathbf{x}, t) = \mathbf{u}(\mathbf{x} - (t - s)\mathbf{u}(\mathbf{x}, t), s)$ for $s \leq t$. Intuitively, if we want to know the velocity $\mathbf{u}(\mathbf{x}, t)$ at point $\mathbf{x}$ at time t, looking back along the characteristic passing through $\mathbf{x}$ at time t to any previous time s; however, the characteristic is the straight line defined by the velocity $\mathbf{u}(\mathbf{x}, t)$ that we want to know. Hence taking an implicit approach to the solution of this equation, which when combined with the operator splitting amounts to

$$\frac{\mathbf{w} - \tilde{\mathbf{u}}^n}{\Delta t} = \mathbf{0} \tag{6.8}$$

$$\rho \frac{\mathbf{u}^{n+1} - \mathbf{w}}{\Delta t} = -\nabla p^{n+1} + \rho \mathbf{g} \tag{6.9}$$

$$\nabla \cdot \mathbf{u}^{n+1} = \mathbf{0} \tag{6.10}$$

using the notation $\mathbf{u}^{n+\alpha}(\mathbf{x}) = \mathbf{u}(\mathbf{x}, t^{n+\alpha})$, $\alpha = 0, 1$ to denote the time $t^{n+\alpha}$ velocities. Furthermore, the intermediate velocity $\mathbf{w}$ is related to $\tilde{\mathbf{u}}^n$ through $\tilde{\mathbf{u}}^n(\mathbf{x}) = \mathbf{u}(\mathbf{x} - \Delta t \mathbf{w}(\mathbf{x}), t^n)$.

6.2.1 Spatial Discretization

The spatial discretization is done by representing the velocity with mulitquadratic B-splines and pressures with multilinear B-splines.

A regular grid with spacing Δx and define pressure degrees of freedom at grid vertices and velocity degrees of freedom at grid cell centers as in [151] (see Figure 6.1) is used.

This efficiently aligns the support of the multiquadratic and multilinear interpolating functions which naturally allows for a grid-cell-wise definition of the flow domain (see Figure 6.4).

The discretization of the velocity and pressure are defined as

$$N_{\mathbf{i}}(\mathbf{x}) = \prod_{\alpha} \hat{N}(\frac{x_\alpha - x_{\alpha\mathbf{i}}}{\Delta x}),\ \chi_{\mathbf{c}}(\mathbf{x}) = \prod_{\alpha} \hat{\chi}(\frac{x_\alpha - x_{\alpha\mathbf{c}}}{\Delta x}) \tag{6.11}$$

$$\hat{N}(\eta) = \begin{cases} \frac{\left(\eta+\frac{3}{2}\right)^2}{2}, & \eta \in (-\frac{3}{2}, -\frac{1}{2}) \\ -\eta^2 + \frac{3}{4}, & \eta \in [-\frac{1}{2}, \frac{1}{2}] \\ \frac{\left(\eta-\frac{3}{2}\right)^2}{2}, & \eta \in (\frac{1}{2}, \frac{3}{2}) \\ 0, & \text{otherwise} \end{cases} \tag{6.12}$$

$$\hat{\chi}(\nu) = \begin{cases} 1+\nu, & \nu \in (-1, 0) \\ 1-\nu, & \nu \in [0, 1) \\ 0, & \text{otherwise} \end{cases} \tag{6.13}$$

where α indicates the components of the vectors $\mathbf{x}$, $\mathbf{x_i}$ and $\mathbf{x_c}$. The velocity and pressure fields become

$$\mathbf{u}(\mathbf{x}) = \sum_{\mathbf{i}} \bar{\mathbf{u}}_{\mathbf{i}} N_{\mathbf{i}}(\mathbf{x}),\ p(\mathbf{x}) = \sum_{\mathbf{c}} p_{\mathbf{c}} \chi_{\mathbf{c}}(\mathbf{x}). \tag{6.14}$$

The notation $\bar{\mathbf{u}}_{\mathbf{i}}$ is used to distinguish it from the velocity at the grid node $\mathbf{u}(\mathbf{x_i}) = \sum_{\mathbf{j}} \bar{\mathbf{u}}_{\mathbf{j}} N_{\mathbf{j}}(\mathbf{x_i})$ since the multiquadratic B-splines are not interpolatory and these will in general be different. Note that multilinear interpolation is interpolatory and $p_{\mathbf{c}} = \sum_{\mathbf{D}} p_{\mathbf{D}} \chi_{\mathbf{D}}(\mathbf{x_c})$.

6.2.2 BSLQB Advection

With this interpolation choice, the intermediate grid node velocity values can be solved$\mathbf{w}(\mathbf{x_i})$ from Equation (6.8) as

$$\mathbf{w}(\mathbf{x_i}) = \sum_{\mathbf{j}} \bar{\mathbf{u}}_{\mathbf{j}}^n N_{\mathbf{j}} \left(\mathbf{x_i} - \Delta t \mathbf{w}(\mathbf{x_i})\right). \tag{6.15}$$

Newton's method can be used to solve the system since the multiquadratic B-splines are C^1. Using $\mathbf{w}_{\mathbf{i}}^k$ to denote the k^{th} Newton approximation to $\mathbf{w}(\mathbf{x_i})$. Explicit semi-Lagrangian is used as an initial guess with $\mathbf{w}_{\mathbf{i}}^0 = \sum_{\mathbf{j}} \bar{\mathbf{u}}_{\mathbf{j}}^n N_{\mathbf{j}} \left(\mathbf{x_i} - \Delta t \sum_{\mathbf{l}} \bar{\mathbf{u}}_{\mathbf{l}}^n N_{\mathbf{l}}(\mathbf{x_i})\right)$ and then updating

iteratively via $\mathbf{w}_\mathbf{i}^k \mathrel{+}= \boldsymbol{\delta}\mathbf{u}^k$ with Newton increment $\boldsymbol{\delta}\mathbf{u}^k$ satisfying

$$\boldsymbol{\delta}\mathbf{u}^k = \left(\mathbf{I} + \Delta t \frac{\partial \mathbf{u}^n}{\partial \mathbf{x}}\left(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{w}_\mathbf{i}^k\right)\right)^{-1} \left(\sum_\mathbf{j} \bar{\mathbf{u}}_\mathbf{j}^n N_\mathbf{j}\left(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{w}_\mathbf{i}^k\right) - \mathbf{w}_\mathbf{i}^k\right) \tag{6.16}$$

where $\frac{\partial \mathbf{u}^n}{\partial \mathbf{x}}\left(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{w}_\mathbf{i}^k\right) = \sum_\mathbf{j} \bar{\mathbf{u}}_\mathbf{j}^n \frac{\partial N_\mathbf{j}}{\partial \mathbf{x}}\left(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{w}_\mathbf{i}^k\right)$. It is generally observed [152, 118] that with BSL approaches of this type, this iteration will converge as long as $\mathbf{I} + \Delta t \sum_\mathbf{j} \bar{\mathbf{u}}_\mathbf{j}^n \frac{\partial N_\mathbf{j}}{\partial \mathbf{x}}\left(\mathbf{x}_\mathbf{i} - \Delta t \mathbf{w}_\mathbf{i}^k\right)$ is non-singular. This condition holds as long as no shocks form under Burgers' equation [98] (forward from time t^n). For incompressible flow shock formation does not occur, but for compressible flows it may be a problem.

In practice, this iteration converges in 3 or 4 iterations, even with CFL numbers larger than 4 . When it does fail (which occurs less than one percent of the time in the examples runned), it is usually for points near the boundary with characteristics that leave the domain (since we cannot estimate $\frac{\partial \mathbf{u}^n}{\partial \mathbf{x}}$ using grid interpolation if the upwind estimate leaves the grid). In this case explicit semi-Lagrangian and boundary conditions are used if the characteristic point is off the domain.

Once the grid node values of the intermediate velocity $\mathbf{w}(\mathbf{x}_\mathbf{i})$ are evaluated, the interpolation coefficients $\bar{\mathbf{w}}_\mathbf{j}$ are determined such that $\mathbf{w}(\mathbf{x}_\mathbf{i}) = \sum_\mathbf{j} \bar{\mathbf{w}}_\mathbf{j} N_\mathbf{j}(\mathbf{x}_\mathbf{i})$.

On the boundary of the grid, setting $\bar{\mathbf{w}}_\mathbf{j} = \mathbf{w}(\mathbf{x}_\mathbf{j})$ since interpolation can only be done $\mathbf{x}_\mathbf{i}$ if all of its neighbors have data.

This yields a square, symmetric positive definite system of equations for the remaining $\bar{\mathbf{w}}_\mathbf{j}$. The system is very well conditioned with sparse, symmetric matrix $N_\mathbf{j}(\mathbf{x}_\mathbf{i})$ consisting of non-negative entries and rows that sum to one. The sparsity and symmetry of the system arises from the compact support and geometric symmetry, respectively, of the B-spline basis functions $N_\mathbf{j}$. The system can be solved to a residual of machine precision in one iteration of PCG (or tens of iterations of unpreconditioned CG).

Determining the coefficients $\bar{\mathbf{w}}_\mathbf{j}$ can lead to increasingly oscillatory velocity fields. This is perhaps due to the unfavorable filtering properties of even order B-splines [120, 121].

However, a simple stabilization strategy can be obtained as

$$\sum_{\mathbf{j}} \left(\lambda N_{\mathbf{j}}(\mathbf{x_i}) + (1-\lambda)\delta_{\mathbf{ij}}\right) \bar{\mathbf{w}}_{\mathbf{j}} = \mathbf{w}(\mathbf{x_i}) \tag{6.17}$$

where $\lambda \in [0, 1]$ and $\delta_{\mathbf{ij}}$ is the Kronecker delta. A value of $\lambda = 0$ is very stable, but extremely dissipative. Stable yet energetic behavior is achieved by decreasing the value of λ under grid refinement. In practice $\lambda \in (.95, 1]$ with $\lambda = c\Delta x$ for constant c provided a good balance without compromising second order accuracy of the method. Noting that Riishøjgaard et al. [123] also added diffusion to cubic spline interpolation based semi-Lagrangian to reduce noise.

6.2.3 Hybrid BSLQB-PolyPIC Advection

In some portions of the domain, we store particles with positions $\mathbf{x}_p^n$ and PolyPIC [4] velocity coefficients $\mathbf{c}_p^n$. In the vicinity of the particles, PolyPIC [4] is used to update the intermediate velocity field $\bar{\mathbf{w}}_{\mathbf{j}}$. First the particle positions is updated as $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^n$ (where the velocity $\mathbf{v}_p^n$ is determined from $\mathbf{c}_p^n$ following [4]). Then the components $\bar{w}_{\mathbf{j}\alpha}$ of the coefficients $\bar{\mathbf{w}}_{\mathbf{j}}$ are determined as

$$\bar{w}_{\mathbf{j}\alpha} = \frac{\sum_p m_p N_{\mathbf{j}}(\mathbf{x}_p^{n+1}) \left(\sum_{r=1}^{N_r} s_r(\mathbf{x_j} - \mathbf{x}_p^{n+1}) c_{pr\alpha}^n\right)}{\sum_p m_p N_{\mathbf{j}}(\mathbf{x}_p^{n+1})} \tag{6.18}$$

where N_r is the number of polynomial modes $s_r(\mathbf{x})$, as in Fu et al. [4]. To create our hybrid approach, $\bar{w}_{\mathbf{j}\alpha}$ is updated from Equation (6.18) whenever the denominator is greater than a threshold $\sum_p m_p N_{\mathbf{j}}(\mathbf{x}_p^{n+1}) > \tau^m$, otherwise BSLQB is used to update from Equation (6.17). The threshold is utilized due to the grid node update in Equation (6.18) loses accuracy when the denominator is near zero and in this case the BSLQB approximation is likely more accurate. The polynomial mode coefficients for the next time step $\mathbf{c}_p^{n+1}$ are determined from the grid velocities at the end of the time step (using particle positions $\mathbf{x}_p^{n+1}$ and after pressure projection).

6.2.4 Pressure Projection

Equations (6.9)- (6.10) and boundary condition Equations (6.6)-6.7 are solved in a variational way. This requires that the dot products of Equations (6.9), (6.10) and Equations (6.6) with arbitrary test functions $\mathbf{r}$, q and μ respectively integrated over the domain are always equal to zero. The free surface boundary condition in Equation (6.7) is satisfied by Equation (6.9).

$$\int_{\Omega} \mathbf{r} \cdot \rho \left(\frac{\mathbf{u}^{n+1} - \mathbf{w}}{\Delta t} \right) d\mathbf{x} = \int_{\Omega} p^{n+1} \nabla \cdot \mathbf{r} + \rho \mathbf{r} \cdot \mathbf{g} d\mathbf{x} - \int_{\partial\Omega} p^{n+1} \mathbf{r} \cdot \mathbf{n} ds(\mathbf{x}) \tag{6.19}$$

$$\int_{\Omega} q \nabla \cdot \mathbf{u}^{n+1} d\mathbf{x} = 0 \tag{6.20}$$

$$\int_{\partial\Omega_D} \mu \left(\mathbf{u}^{n+1} \cdot \mathbf{n} - a \right) ds(\mathbf{x}) = 0. \tag{6.21}$$

Integration by parts is used Equation (6.9).

Furthermore, $\int_{\partial\Omega} p^{n+1} \mathbf{r} \cdot \mathbf{n} ds(\mathbf{x})$ in Equation6.19 is modified with the boundary conditions.

The pressure is zero on $\partial\Omega_N$, however the pressure is unknown on $\partial\Omega_D$. Introducing a pressure Lagrange multiplier λ^{n+1} associated with satisfing the velocity boundary condition in Equation (6.21). This represents the external pressure needed in $\partial\Omega_D$ to ensure that $\mathbf{u}^{n+1} \cdot \mathbf{n} = a$. With this convention, $\int_{\partial\Omega} p^{n+1} \mathbf{r} \cdot \mathbf{n} ds(\mathbf{x}) = \int_{\partial\Omega_D} \lambda^{n+1} \mathbf{r} \cdot \mathbf{n} ds(\mathbf{x})$. Unlike Equation (6.21) (and its strong form 6.6) that requires introduction of a Lagrange multiplier, Equation (6.7) is enforced through the weak form by setting $p^{n+1} = 0$ in the integral over $\partial\Omega_N$ in Equation (6.19).

Discretization in space requires the test functions interpolations $\mathbf{r}$, q and μ. Using the same spaces as in Equation (6.14) for velocity and pressure for $\mathbf{r} = \sum_{\mathbf{i}} \bar{\mathbf{r}}_{\mathbf{i}} N_{\mathbf{i}}$ and $q = \sum_{\mathbf{D}} q_{\mathbf{D}} \chi_{\mathbf{D}}$.

For the test functions μ, we choose the same space as q, p, but with functions restricted to $\partial\Omega_D$, $\mu = \sum_{\mathbf{B}} \mu_{\mathbf{B}} \chi_{\mathbf{B}}$ for $\mathbf{B}$ with grid cell $\Omega_{\mathbf{B}} \cap \partial\Omega_D \neq \emptyset$ (see Figure 6.4). We choose the same space for $\lambda^{n+1} = \sum_{\mathbf{B}} \lambda_{\mathbf{B}}^{n+1} \chi_{\mathbf{B}}$ to close the system. With these choices for the test

functions, the variational problem is projected to a finite dimensional problem defined by the interpolation degrees of freedom. This is expressed as a linear system for velocities $\bar{\mathbf{u}}_{\mathbf{j}}^{n+1}$, internal pressures $p_{\mathbf{c}}^{n+1}$, and external pressures $\lambda_{\mathbf{B}}^{n+1}$ that is equivalent to

$$\begin{pmatrix} \mathbf{M} & -\mathbf{D}^T & \mathbf{B}^T \\ -\mathbf{D} & & \\ \mathbf{B} & & \end{pmatrix} \begin{pmatrix} \mathbf{U}^{n+1} \\ \mathbf{P}^{n+1} \\ \mathbf{\Lambda}^{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{MW} + \hat{\mathbf{g}} \\ \mathbf{0} \\ \mathbf{A} \end{pmatrix}. \tag{6.22}$$

$\mathbf{U}^{n+1}$, $\mathbf{P}^{n+1}$ and $\mathbf{\Lambda}^{n+1}$ are the vectors of all unknown $\bar{\mathbf{u}}_{\mathbf{j}}^{n+1}$, $p_{\mathbf{c}}^{n+1}$ and $\lambda_{\mathbf{B}}^{n+1}$ respectively. Furthermore, $\mathbf{M}$ is the mass matrix, $\mathbf{B}$ defines the velocity boundary conditions and $\mathbf{D}$ defines the discrete divergence condition. Lastly, $\mathbf{W}$ is the vector of all $\bar{\mathbf{W}}_{\mathbf{i}}$ that define the intermediate velocity, $\hat{\mathbf{g}}$ is from gravity and $\mathbf{A}$ is the variational boundary condition. Using the convention that Greek indices α, β range from $1-3$, these matrices and vectors have entries

$$M_{\alpha\mathbf{i}\beta\mathbf{j}} = \delta_{\alpha\beta} \int_\Omega \frac{\rho}{\Delta t} N_{\mathbf{i}} N_{\mathbf{j}} d\mathbf{x} \tag{6.23}$$

$$D_{\mathbf{d}\beta\mathbf{j}} = \int_\Omega \chi_{\mathbf{d}} \frac{\partial N_{\mathbf{j}}}{\partial x_\beta} d\mathbf{x} \tag{6.24}$$

$$\hat{g}_{\alpha\mathbf{i}} = \int_\Omega \rho g_\alpha N_{\mathbf{i}} d\mathbf{x} \tag{6.25}$$

$$B_{\mathbf{b}\beta\mathbf{j}} = \int_{\Omega_D} \chi_{\mathbf{b}} N_{\mathbf{j}} n_\beta ds(\mathbf{x}) \tag{6.26}$$

$$A_{\mathbf{b}} = \int_\Omega a \chi_{\mathbf{b}} ds(\mathbf{x}). \tag{6.27}$$

If $\mathbf{g} = [-\mathbf{D}^T, \mathbf{B}^T]$, this system becomes a symmetric positive definite one for $\mathbf{P}^{n+1}$ and $\mathbf{\Lambda}^{n+1}$ followed by a velocity correction for $\mathbf{u}^{n+1}$

$$\begin{pmatrix} \mathbf{P}^{n+1} \\ \mathbf{\Lambda}^{n+1} \end{pmatrix} = \left(\mathbf{g}^T \mathbf{M}^{-1} \mathbf{g}\right)^{-1} \left(\mathbf{g}^T \left(\mathbf{W} + \mathbf{M}^{-1} \hat{\mathbf{g}} \right) - \begin{pmatrix} \mathbf{0} \\ \mathbf{A} \end{pmatrix} \right) \tag{6.28}$$

$$\mathbf{u}^{n+1} = -\mathbf{M}^{-1} \mathbf{g} \begin{pmatrix} \mathbf{P}^{n+1} \\ \mathbf{\Lambda}^{n+1} \end{pmatrix} + \mathbf{W} + \mathbf{M}^{-1} \hat{\mathbf{g}}. \tag{6.29}$$

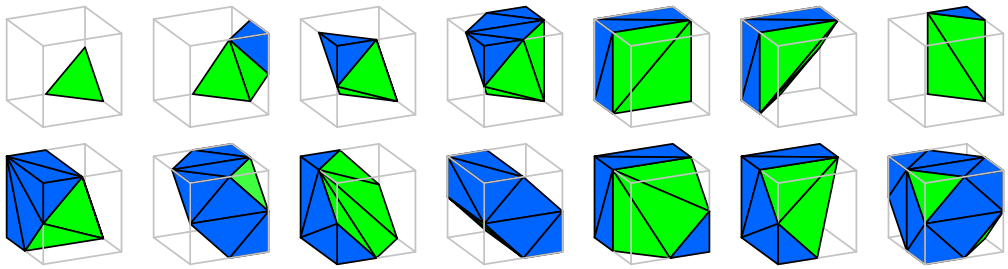


Figure 6.3: **Cut-Cells**. We show the 14 essential cases used in determining the cut-cell fluid domain geometry. Blue faces indicate the intersection of the grid cell with the fluid domain. Green faces indicate the velocity boundary condition faces on $\partial\Omega_D$.

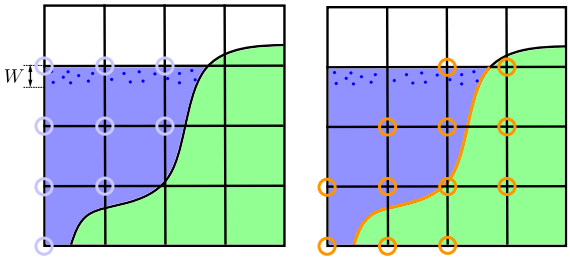


Figure 6.4: **Boundary Lagrange Multiplier**. **Left**: We define the fluid domain to consist of cells that either have (1) a particle (dark blue) in it or (2) a node with non-positive level set value (light blue). **Right**: Boundary Lagrange multiplier external pressure $\lambda_{\mathbf{b}}$ (orange circles) are like the interior pressures $p_{\mathbf{c}}$ except only defined on fluid domain cells that intersect $\partial\Omega_D$.

Unfortunately, this system will be dense in the current formulation since the full mass matrix $M_{\alpha\mathbf{i}\beta\mathbf{j}}$ is non-diagonal with dense inverse [150]. However, a simple lumped mass approximation

$$M^l_{\alpha\mathbf{i}\beta\mathbf{j}} = \begin{cases} \delta_{\alpha\beta}\int_\Omega \frac{\rho}{\Delta t} N_{\mathbf{i}} d\mathbf{x}, & \mathbf{i} = \mathbf{j} \\ 0, & \text{otherwise} \end{cases} \tag{6.30}$$

gives rise to a sparse matrix in Equation (6.28).

6.2.5 Cut-Cells

As in XFEM and VNA approaches [143, 144, 104], we resolve sub-grid-cell geometry by simply performing the integrations in Equations (6.25)-6.27 over the geometry of the fluid domain. We use a level set to define solid boundaries (green in Figure 6.4) on which velocity boundary conditions are defined. We triangulate the zero isocontour using marching cubes [68] (see Figure 6.3). The integrals in Equations (6.25)- (6.27) all involve polynomials over volumetric polyhedra (Equations (6.25), blue in Figure 6.3) or surface polygons (Equations (6.27), green in Figure 6.3) and we use Gauss quadrature of order adapted to compute the integrals with no error (see [153]). For free surface flows, we use particles to denote grid cells with fluid in them. Cells near the solid boundary are clipped by the marching cubes geometry. The fluid domain Ω is defined as the union of all clipped and full fluid cells (see Figure 6.4).

6.3 BSLQB Comparison with Explicit Semi-Lagrangian

We demonstrate improved resolution of flow detail with BSLQB compared to explicit semi-Lagrangian in a 2D example of smoke flowing past a circle (see Figure 6.5) and with a 2D spinning circle example (see Figure 6.6). Note that particles are only used for flow visualization and not for PolyPIC advection in these examples. BSLQB exhibits more energetic, turbulent flows than semi-Lagrangian advection. Notably, the BSLQB result breaks symmetry sooner. In Figure 6.5 we also examine the effect of extremal values of the λ parameter described in Equation (6.17). A zero value of λ is quite dissipative compared to a full value of $\lambda = 1$ for both semi-Lagrangian and BSLQB. As mentioned in Section 6.2.2, we generally found that keeping λ close to 1 provided the least dissipative behavior, while setting the value slightly less than 1 helped restore stability when necessary (one can also dynamically adjust this value over the course of a simulation). In Figure 6.6, we initially set the angular

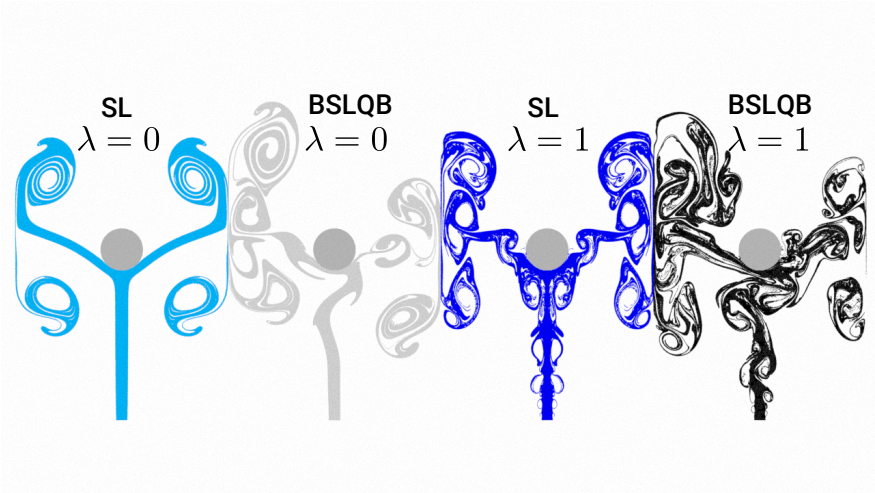
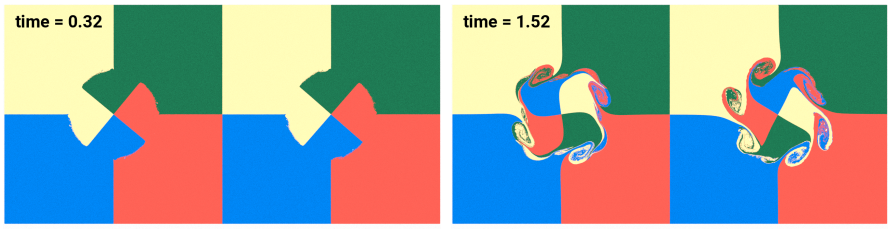


Figure 6.5: **2D Smoke Pass Circle**. BSLQB exhibits more fine-scale flow detail and vorticity than semi-Lagrangian for extremal values of interpolation parameter λ (Equation (6.17)). From left to right: semi-Lagrangian with $\lambda = 0$, BSLQB with $\lambda = 0$, semi-Lagrangian with $\lambda = 1$, BSLQB with $\lambda = 1$.



velocity to 4 radians per second in a circle of radius .2 (with $\Omega = [0, 1] \times [0, 1]$). The simulation is run with $\Delta x = \frac{1}{511}$ and a $\Delta t = .02$ (CFL number of 3).

We examine the convergence behavior of BSLQB for the 2D Burgers' equation $\frac{D\mathbf{u}}{Dt} = \mathbf{0}$ with initial data $\mathbf{u}(\mathbf{x}) = \mathbf{x} \cdot (\mathbf{A}\mathbf{x})$ for $\mathbf{A} = \mathbf{r}\mathbf{\Lambda}\mathbf{r}^T$ for diagonal $\mathbf{\Lambda}$ with entries 1 and .25 and rotation (of .1 radians) $\mathbf{r}$ (see Figure 6.7). We examine the convergence behavior under refinement in space and time with $\Delta t = \Delta x$. We compute the best fit line to the plot of the logarithm of the L^∞ norm of the error versus the logarithm of Δx for a number of grid

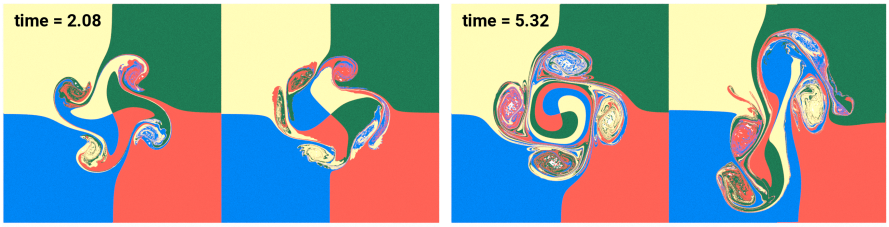


Figure 6.6: **2D Inner Rotation SL vs. BSLQB**. We compare semi-Lagrangian (left) and BSLQB (right) in a vorticity-intensive example. BSLQB breaks symmetry and exhibits a more turbulent flow pattern. Note we only use particles for flow visualization and not for PolyPIC advection in this example.

resolutions. We observe slopes of approximately 2 for BSLQB with interpolation parameter $\lambda = 1$ and $\lambda = 1 - c\Delta x$ (with $c = 2.95$), indicating second order accuracy in space and time under refinement. We observe slopes of approximately 1 for explicit semi-Lagrangian, indicating first order.

6.4 Results

6.4.1 Hybrid BSLQB/PolyPIC

We demonstrate our hybrid BSLQB/PolyPIC advection with water simulation. We prevent excessive run times by utilizing a narrow band of particles near the free surface and a level set (with BSLQB advection) in deeper levels. Figure 6.8 Top shows a disc of water splashing in a rectangular tank with dimension 1×2 and grid cell size $\Delta x = 1/255$. The time step is restricted to be in the range $\Delta t \in [0.005, 0.01]$. 20 particles are initialized in every cell that is initially in a narrow band of $7\Delta x$ below the zero isocontour of the level set. Figure 6.8 Bottom shows an analogous 3D example where a sphere of water splashes in a tank. A cell size of $\Delta x = \frac{1}{63}$ is used in a domain with dimensions $1 \times 2 \times 1$. We take a fixed time step of $\Delta t = 0.01$ and demonstrate that narrow banding does not prevent larger-than-CFL time

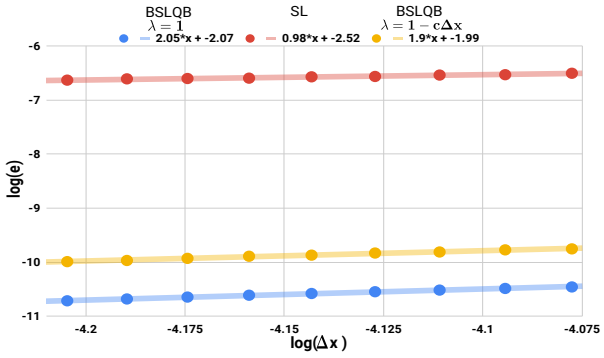


Figure 6.7: **SL vs BSLQB Convergence**: We compare explicit semi-Lagrangian (SL, red), with BSLQB (blue) and interpolation coefficient $\lambda = 1$ (Equation (6.17)) and BSLQB with interpolation coefficient $\lambda = 1 - c\Delta x$ (orange). We plot $\log(\Delta x)$ versus $\log(e)$ (where e is the infinity norm of the error) for a variety of grid resolutions Δx and compute the best fit lines. The slope of the line provides empirical evidence for the convergence rate of the method.

steps. 1,008,187 particles are used to resolve the free surface in a narrow band of width $5\Delta x$. As in 2D, the particles capture highly-dynamic behavior of the free surface while the level set is sufficient to represent the bulk fluid in the bottom half of the domain.

6.4.2 Cut-Cell Examples

We demonstrate the ability of our cut-cell method to produce detailed flows in complicated irregular domains for smoke and free surface water examples. Figure 6.10 demonstrates the subtle and visually interesting behavior that arises as smoke flow to the center of a cubic domain colliding with a spherical boundary. We consider dam break simulations in rectangular domains with a bunny obstacle (Figure 6.9).This example use a grid cell size of $\Delta x = 1/127$, 8 particles per cell and a fixed time step of $\Delta t = 0.003$. In this example, the water accurately conforms to the irregular domain from the obstacle modeled with cut-cells.

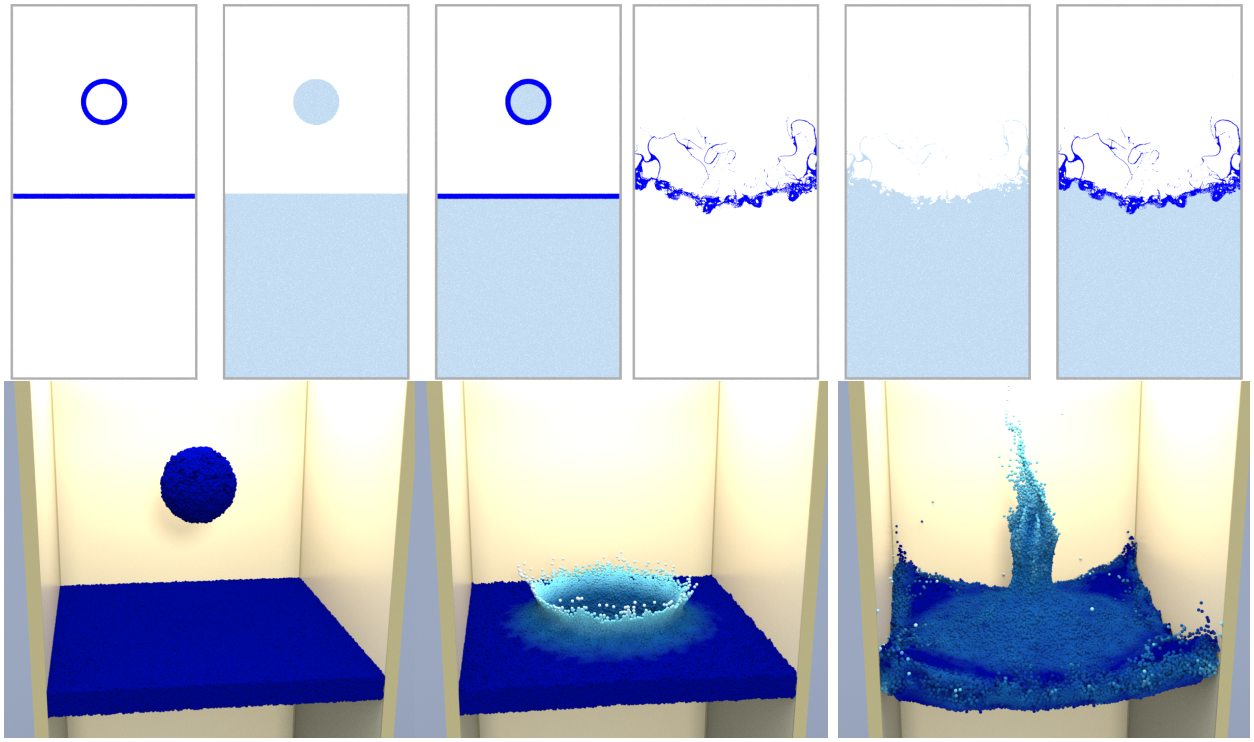


Figure 6.8: **Narrow Band Free Surface**. A circle/sphere falls in a tank of water under gravity. Using only a narrow band of particles saves computational cost and enables increased resolution of the free surface. **Top**: In 2D we illustrate the hybrid particle(dark blue)/level set (light blue) representation. **Bottom**: Particles are colored based on velocity magnitude.

6.4.3 Performance Considerations

The implementation of our method takes advantage of hybrid parallelism (MPI, OpenMP, and CUDA/OpenCL) on heterogeneous compute architectures in order to achieve practical runtime performance (see Table 6.1 for 3D example performance numbers). The spatial domain is uniformly divided into subdomains assigned to distinct MPI ranks, which distributes much of the computational load at the expense of synchronization overhead exchanging ghost information across ranks. On each rank, steps of our time integration loop such as BSLQB advection are multithreaded using OpenMP or CUDA when appropriate. The dominant costs per time step are the solution of the pressure projection system and, in the case of

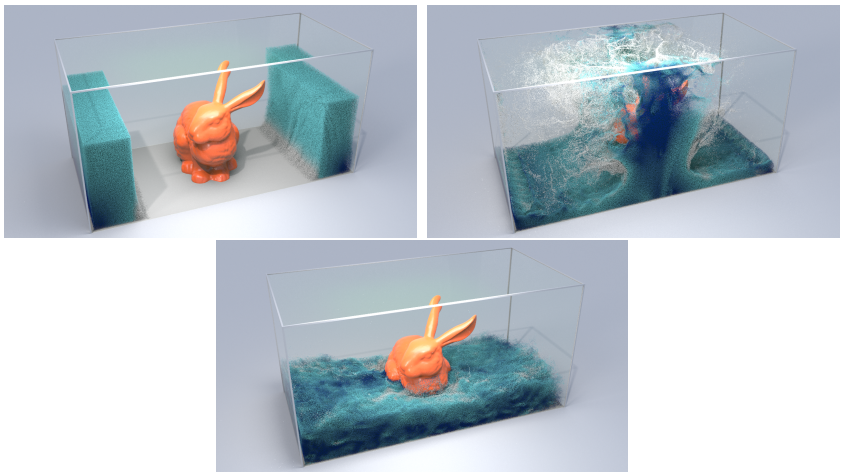


Figure 6.9: **Dam Break with Bunny**. Opposing blocks of water collapse in a tank and flow around the irregular domain boundary placed in the middle of the tank. Particles are colored from slow (blue) to fast (white) speed.

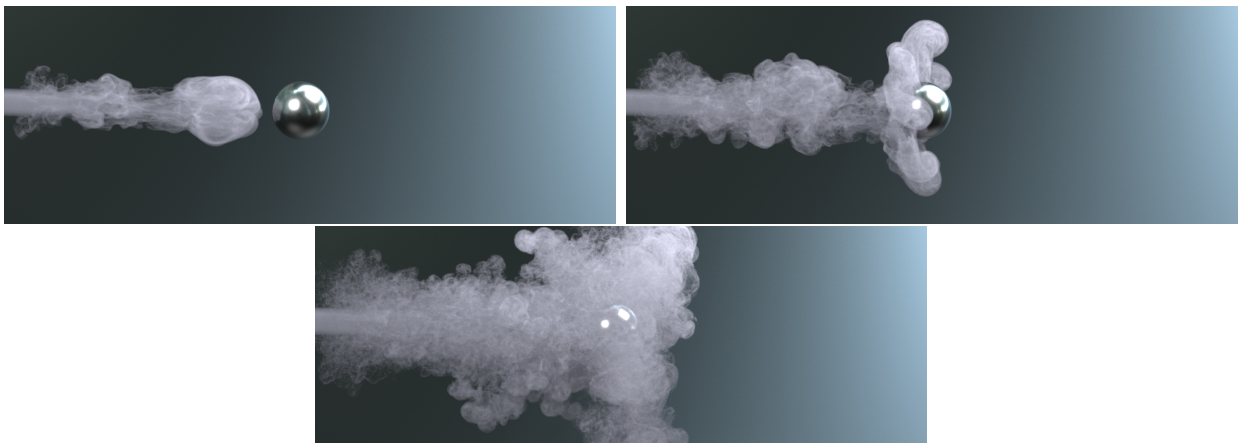


Figure 6.10: **Smoke Jet**. A plume of smoke is simulated with BSLQB. Zero normal velocity boundary conditions are enforced on the irregular boundary of the sphere inducing intricate flow patterns as the smoke approaches it.

free surface simulation, assembly of the pressure system and its preconditioner. We permute Equation (6.28) so that each rank's degrees of freedom are contiguous in the solution vector then solve the system using AMGCL [154] using the multi-GPU VexCL backend (or the OpenMP CPU backend on more limited machines). Using a strong algebraic multigrid

preconditioner with large-degree Chebyshev smoothing allows our system to be solved to desired tolerance in tens of iterations, even at fine spatial resolution. An important step in minimizng the cost of system assembly is to scalably parallelize sparse matrix-matrix multiplication, for which we use the algorithm of Saad [155]. In the future, we are interested in implementing load balancing strategies such as the simple speculative load balancing approach of [156], particularly for free surface flows. We note that our implementation enables high-resolution simulations such as that in Figure 6.11 at relatively modest computational cost (see Table 6.1).

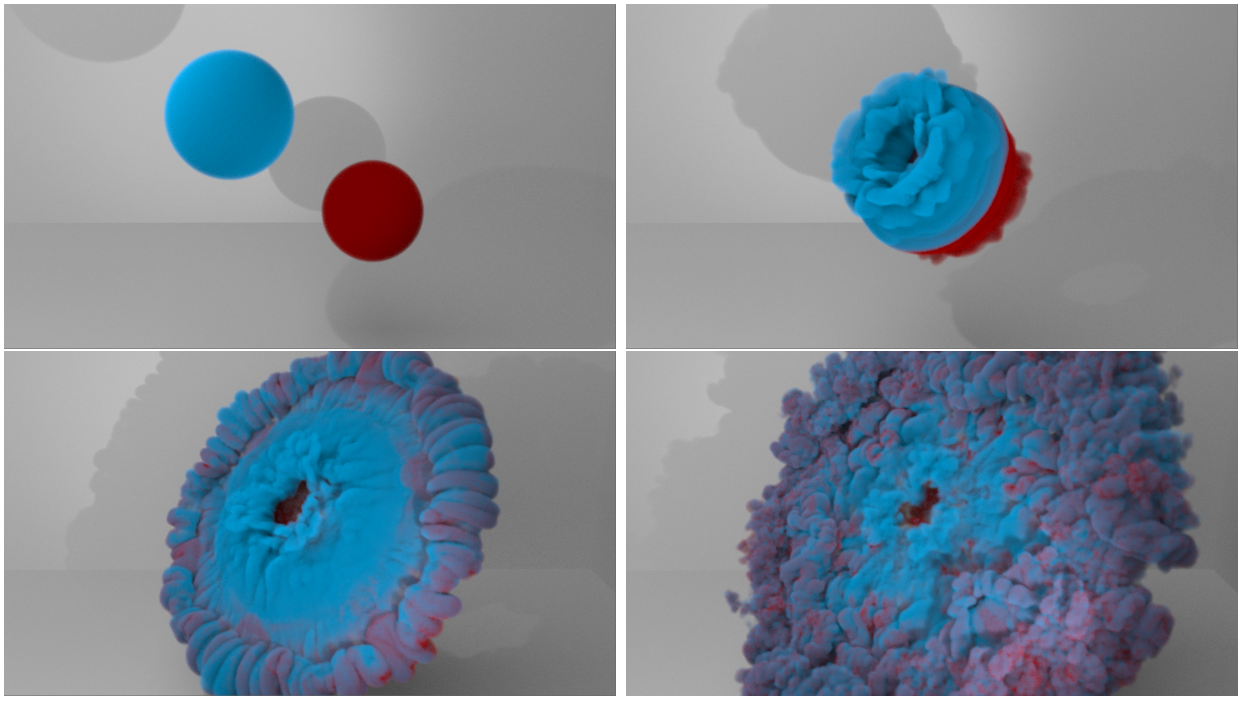


Figure 6.11: **High-Resolution Smoke**. Two spheres of smoke collide in a high-resolution 3D simulation ($\Delta x = 1/255$). BSLQB accurately resolves vorticial flow detail.

Example	Seconds	# Particles	Δx^{-1}
Smoke Jet (Fig. 6.10)	1,212	12,502,349	127
Smoke Spheres* (Fig. 6.11)	428	64,000,000	255
Narrow Band (Fig. 6.8)	396	1,008,187	63
Bunny Dam Break (Fig. 6.9)	1,171	4,797,535	127

Table 6.1: Average time per frame (in seconds) for each of the 3D examples shown in the paper. Examples were run on workstations with 16-core CPUs running at 2.20 GHz, except for the smoke spheres example, which was run on a cluster equipped with CPUs running at 3.07 GHz and Nvidia Tesla V100 GPUs which were used for the linear solves.

6.5 Limitations and Future work

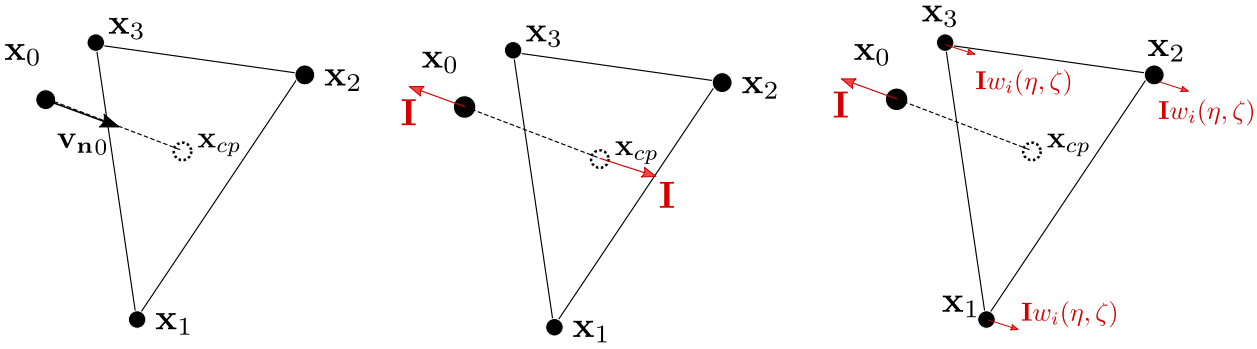
Our approach has several key limitations that could be improved. First, our adoption of collocated multiquadratic velocity and multilinear pressure is a significant departure from most fluid solvers utilized in graphics applications. We note that BSLQB and BSLQB/PolyPIC could be used with a MAC grid; however, each velocity face component would have to be solved for individually. Another drawback for our multiquadratic velocity and multilinear pressure formulation is that it gives rise to a very wide pressure system stencil consisting of 49 non-zero entries per row in 2D and 343 in 3D. Collocated approaches that make use of multilinear velocities and constant pressure give rise to 9 (2D) and 27 (3D) entries per row [157], however they do not allow for C^1 continuity and require spurious pressure mode damping. Our wide stencils likely negatively affect the efficacy of preconditioning techniques as well, however we were very pleased with the efficiency of the AMGCL [154] library. Also, while the use of mass lumping in Equation (6.30) is necessary to ensure a sparse pressure projection system, Boatella et al. [150] note that this has been shown to degrade accuracy. In fact, Boatela et al. [150] introduce a sparse approximate inverse to the full mass matrix to avoid dense systems of equations in the pressure solve without degrading accuracy. Split

cubic interpolation, which approximates similar systems with tridiagonal ones could also possibly be used for this [158]. Adoption of one of these approaches with our formulation would be an interesting area of future work. Also, we note that the more sophisticated transition criteria for narrow banding techniques in Sato et al. [107] could naturally be used with our method. Finally, we note that the work of Zehnder et al. [136, 137] could be easily applied to our technique to further reduce dissipation since it is based on the Chorin [84] splitting techniques (Equations (6.8)-(6.10)) that we start from.

APPENDIX A

Momentum Conservation of Impulses

A.1 Point-Triangle pair impulses



Consider a point with mass m_0 and coordinate x_0 in a collision course with a triangle defined by vertices with masses m_1, m_2, m_3 and coordinates x_1 , x_2 , x_3.

The initial velocities of the points are v_0 , v_1 , v_2 , v_3. The closest point and the normal from the triangle of the closest point can be defined as follows:

$$\underline{x}_{cp} = w_1\underline{x}_1 + w_2\underline{x}_2 + w_3\underline{x}_3 \tag{A.1}$$

$$\underline{n} = \underline{x}_0 - \underline{x}_{cp} \tag{A.2}$$

The linear momemtum before are

$$\underline{p}_i = m_i\underline{v}_i \tag{A.3}$$

The linear momemtum after the impulses I are applied

$$\underline{p}_0 = m_0\underline{v}_0 + I\underline{n} \tag{A.4}$$

$$\underline{p}_i = m_i\underline{v}_i - w_i I\underline{n}, i = 1, 2, 3 \tag{A.5}$$

Now we prove that the linear and angular momentum are conserved after the impulses are applied. For linear momentum.

$$\underline{p}_{after} - \underline{p}_{before} = I\underline{n} - (w_1 + w_2 + w_3)I\underline{n} \tag{A.6}$$

$$= (1 - (w_1 + w_2 + w_3))I\underline{n} = \underline{0} \tag{A.7}$$

The angular momentum around the center of mass can be shown as follows:

$$\underline{l}_{after} - \underline{l}_{before} = \underline{r}_0 \times I\underline{n} - \underline{r}_1 \times w_1 I\underline{n} - \underline{r}_2 \times w_2 I\underline{n} - \underline{r}_3 \times w_3 I\underline{n} \tag{A.8}$$

$$= (\underline{r}_0 - \underline{r}_1 w_1 - \underline{r}_2 w_2 - \underline{r}_3 w_3) \times I\underline{n} \tag{A.9}$$

$$= (\underline{x}_0 - \underline{x}_{com} - (\underline{x}_1 - \underline{x}_{com})w_1 - (\underline{x}_2 - \underline{x}_{com})w_2 - (\underline{x}_3 - \underline{x}_{com})w_3) \times I\underline{n} \tag{A.10}$$

$$= (\underline{x}_0 - w_1\underline{x}_1 - w_2\underline{x}_2 - w_3\underline{x}_3) - \underline{x}_{com} + (w_1 + w_2 + w_3)\underline{x}_{com}) \times I\underline{n} \tag{A.11}$$

$$= (\underline{x}_0 - w_1\underline{x}_1 - w_2\underline{x}_2 - w_3\underline{x}_3)) \times I\underline{n} \tag{A.12}$$

$$= (\underline{x}_0 - \underline{x}_{cp}) \times I\underline{n} \tag{A.13}$$

$$= \underline{n} \times I\underline{n} = \underline{0} \tag{A.14}$$

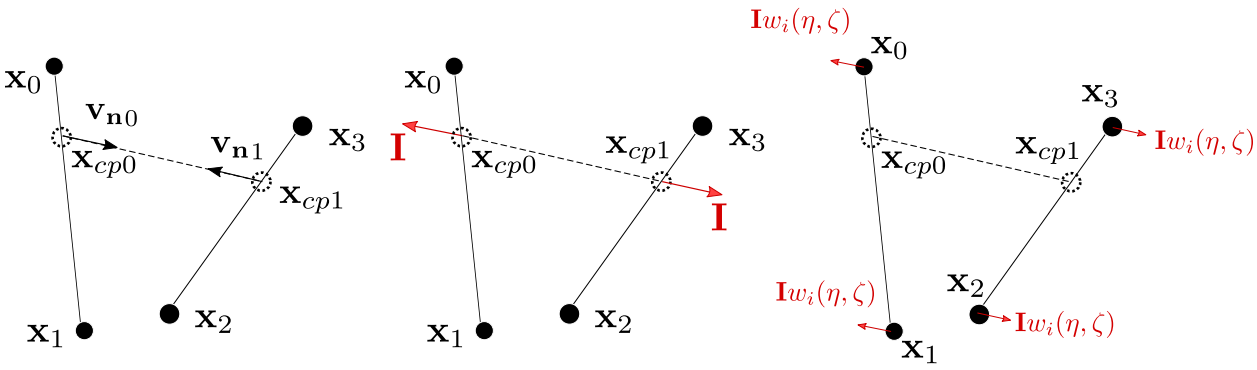
$$\tag{A.15}$$

The impulse can be obtained with the following equation

$$I = -\frac{\underline{n} \cdot (\underline{v}_0 - (w_1\underline{v}_1 + w_2\underline{v}_2 + w_3\underline{v}_3))}{\frac{1}{m_0} + \frac{w_1^2}{m_1} + \frac{w_2^2}{m_2} + \frac{w_3^2}{m_3}} \tag{A.16}$$

A.2 Segment-Segment pair impulses

Consider a segment defined by vertices with masses m_0 and m_1 and coordinates x_0 and x_1 in a collision course with another segment defined by vertices with masses, m_2 and m_3 and



coordinates x_2 and x_3.

The initial velocities of the points are v_0 , v_1 , v_2 , v_3. The closest point for each segments and the normal from each segment closest points can be defined as follows:

$$\underline{x}_{cp0} = w_0\underline{x}_0 + w_1\underline{x}_1 \tag{A.17}$$

$$\underline{x}_{cp1} = w_2\underline{x}_2 + w_3\underline{x}_3 \tag{A.18}$$

$$\underline{n} = \underline{x}_{cp0} - \underline{x}_{cp1} \tag{A.19}$$

The linear momemtum before are

$$\underline{p}_i = m_i\underline{v}_i \tag{A.20}$$

The linear momemtum after the impulses I are applied

$$\underline{p}_i = m_i\underline{v}_i + w_iI\underline{n}, i = 0, 1 \tag{A.21}$$

$$\underline{p}_i = m_i\underline{v}_i - w_iI\underline{n}, i = 2, 3 \tag{A.22}$$

Now we prove that the linear and angular momemtum are conserved after the impulses are applied. For linear momemtum.

$$\underline{p}_{after} - \underline{p}_{before} = (w_0 + w_1)I\underline{n} - (w_2 + w_3)I\underline{n} \tag{A.23}$$

$$= (w_0 + w_1 - (w_2 + w_3))I\underline{n} = \underline{0} \tag{A.24}$$

The angular momentum around the center of mass can be shown as follows:

$$\underline{l}_{after} - \underline{l}_{before} = \underline{r}_0 \times w_0 I\underline{n} + \underline{r}_1 \times w_1 I\underline{n} - \underline{r}_2 \times w_2 I\underline{n} - \underline{r}_3 \times w_3 I\underline{n} \tag{A.25}$$

$$= (\underline{r}_0 w_0 + \underline{r}_1 w_1 - \underline{r}_2 w_2 - \underline{r}_3 w_3) \times I\underline{n} \tag{A.26}$$

$$= ((\underline{x}_0 - \underline{x}_{com})w_0 + (\underline{x}_1 - \underline{x}_{com})w_1 - (\underline{x}_2 - \underline{x}_{com})w_2 - (\underline{x}_3 - \underline{x}_{com})w_3) \times I\underline{n} \tag{A.27}$$

$$= (w_0\underline{x}_0 + w_1\underline{x}_1 - w_2\underline{x}_2 - w_3\underline{x}_3) - (w_0 + w_1)\underline{x}_{com} + (w_2 + w_3)\underline{x}_{com}) \times I\underline{n} \tag{A.28}$$

$$= (w_0\underline{x}_0 + w_1\underline{x}_1 - w_2\underline{x}_2 - w_3\underline{x}_3)) \times I\underline{n} \tag{A.29}$$

$$= (\underline{x}_{cp0} - \underline{x}_{cp1}) \times I\underline{n} \tag{A.30}$$

$$= \underline{n} \times I\underline{n} = \underline{0} \tag{A.31}$$

$$\tag{A.32}$$

The impulse can be obtained with the following equation

$$I = -\frac{\underline{n} \cdot (w_0\underline{v}_0 + w_1\underline{v}_1 - (w_2\underline{v}_2 + w_3\underline{v}_3))}{\frac{w_o^2}{m_0} + \frac{w_1^2}{m_1} + \frac{w_2^2}{m_2} + \frac{w_3^2}{m_3}} \tag{A.33}$$

REFERENCES

[1] E. Love and D. Sulsky, "An unconditionally stable, energy-momentum consistent implementation of the the material point method," *Comp Meth App Mech Eng*, vol. 195, pp. 3903–3925, 2006.

[2] D. Sulsky, S. Zhou, and H. Schreyer, "Application of a particle-in-cell method to solid mechanics," *Comp Phys Comm*, vol. 87, no. 1, pp. 236–252, 1995.

[3] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin, "The affine particle-in-cell method," *ACM Trans Graph*, vol. 34, no. 4, pp. 51:1–51:10, 2015.

[4] C. Fu, Q. Guo, T. Gast, C. Jiang, and J. Teran, "A polynomial particle-in-cell method," *ACM Trans Graph*, vol. 36, pp. 222:1–222:12, Nov. 2017.

[5] M. Tupek, J. Koester, and M. Mosby, "A momentum preserving frictional contact algorithm based on affine particle-in-cell grid transfers," 2021.

[6] A. Stomakhin, R. Howes, C. Schroeder, and J. Teran, "Energetically consistent invertible elasticity," in *Proc Symp Comp Anim*, pp. 25–32, 2012.

[7] H. L. S. Allen R. York II, Deborah Sulsky, "Fluid–membrane interaction based on the material point method," *Int J Numer Meth Eng*, vol. 48, no. 6, 2000.

[8] C. Jiang, C. Schroeder, and J. Teran, "An angular momentum conserving affine-particle-in-cell method," *J Comp Phys*, vol. 338, pp. 137 – 164, 2017.

[9] X. Han, T. F. Gast, Q. Guo, S. Wang, C. Jiang, and J. Teran, "A hybrid material point method for frictional contact with diverse materials," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 2, pp. 1–24, Jul 2019.

[10] R. Bridson, R. Fedkiw, and J. Anderson, "Robust treatment of collisions, contact and friction for cloth animation," *ACM Trans Graph*, vol. 21, no. 3, pp. 594–603, 2002.

[11] D. Levin, J. Litven, G. Jones, S. Sueda, and D. Pai, "Eulerian solid simulation with contact," *ACM Trans Graph*, vol. 30, no. 4, pp. 36:1–36:10, 2011.

[12] Y. Li and J. Barbič, "Immersion of self-intersecting solids and surfaces," *ACM Trans. Graph.*, vol. 37, July 2018.

[13] Y. Teng, D. Levin, and T. Kim, "Eulerian solid-fluid coupling," *ACM Trans Graph*, vol. 35, no. 6, pp. 200:1–200:8, 2016.

[14] K. Muller, D. Fedosov, and G. Gompper, "Smoothed dissipative particle dynamics with angular momentum conservation," *J Comp Phys*, vol. 281, pp. 301–315, 2015.

[15] E. Sifakis, S. Marino, and J. Teran, "Globally coupled collision handling using volume preserving impulses," in *Proc 2008 ACM SIGGRAPH/Eurographics Symp Comp Anim*, pp. 147–153, 2008.

[16] K. Wu and C. Yuksel, "Real-time hair mesh simulation," in *ACM SIGGRAPH Symp Int 3D Graph Games*, ACM, 2016.

[17] E. Sifakis and J. Barbic, "Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction," in *ACM SIGGRAPH 2012 Courses*, SIGGRAPH '12, (New York, NY, USA), pp. 20:1–20:50, ACM, 2012.

[18] M. Arroyo and M. Ortiz, "Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods," *International journal for numerical methods in engineering*, vol. 65, no. 13, pp. 2167–2202, 2006.

[19] T. Brochu, E. Edwards, and R. Bridson, "Efficient geometrically exact continuous collision detection," *ACM Trans Graph*, vol. 31, no. 4, pp. 96:1–96:7, 2012.

[20] W. Zheng, B. Zhu, B. Kim, and R. Fedkiw, "A new incompressibility discretization for a hybrid particle mac grid representation with surface tension," *J Comp Phys*, vol. 280, pp. 96–142, 2015.

[21] N. Thürey, C. Wojtan, M. Gross, and G. Turk, "A multiscale approach to mesh-based surface tension flows," *ACM Trans Graph (TOG)*, vol. 29, no. 4, pp. 1–10, 2010.

[22] C. Wojtan, N. Thürey, M. Gross, and G. Turk, "Physics-inspired topology changes for thin fluid features," *ACM Trans Graph*, vol. 29, no. 4, pp. 50:1–50:8, 2010.

[23] F. Da, D. Hahn, C. Batty, C. Wojtan, and E. Grinspun, "Surface-only liquids," *ACM Trans Graph (TOG)*, vol. 35, no. 4, pp. 1–12, 2016.

[24] S. Yang, X. He, H. Wang, S. Li, G. Wang, E. Wu, and K. Zhou, "Enriching sph simulation by approximate capillary waves," in *Symp Comp Anim*, pp. 29–36, 2016.

[25] W. Li, D. Liu, M. Desbrun, J. Huang, and X. Liu, "Kinetic-based multiphase flow simulation," *IEEE Trans Vis Comp Graph*, 2020.

[26] B. Zhu, E. Quigley, M. Cong, J. Solomon, and R. Fedkiw, "Codimensional surface tension flow on simplicial complexes," *ACM Trans Graph (TOG)*, vol. 33, no. 4, pp. 1–11, 2014.

[27] F. Da, C. Batty, C. Wojtan, and E. Grinspun, "Double bubbles sans toil and trouble: discrete circulation-preserving vortex sheets for soap films and foams," *ACM Trans Graph (SIGGRAPH 2015)*, 2015.

[28] W. Huang, J. Iseringhausen, T. Kneiphof, Z. Qu, C. Jiang, and M. Hullin, "Chemomechanical simulation of soap film flow on spherical bubbles," *ACM Trans Graph*, vol. 39, July 2020.

[29] X. Zhao, S. Xu, and J. Liu, "Surface tension of liquid metal: role, mechanism and application," *Front Energy*, vol. 11, no. 4, pp. 535–567, 2017.

[30] D. Sulsky and A. Kaul, "Implicit dynamics in the material-point method," *Comp Meth in App Mech Eng*, vol. 193, no. 12, pp. 1137–1170, 2004.

[31] M. Müller, "Fast and robust tracking of fluid surfaces," in *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 237–245, ACM, 2009.

[32] T. Brochu, C. Batty, and R. Bridson, "Matching fluid simulation elements to surface geometry and topology," *ACM Trans Graph*, vol. 29, no. 4, pp. 47:1–47:9, 2010.

[33] T. Brochu and R. Bridson, "Robust topological operations for dynamic explicit surfaces," *SIAM J Sci Comp*, vol. 31, no. 4, pp. 2472–2493, 2009.

[34] C. Batty, S. Xenos, and B. Houston, "Tetrahedral embedded boundary methods for accurate and flexible adaptive fluids," in *Comp Graph For*, vol. 29, pp. 695–704, Wiley Online Library, 2010.

[35] F. Sin, A. Bargteil, and J. Hodgins, "A point-based method for animating incompressible flow," in *Proc 2009 ACM SIGGRAPH/Eurographics Symp Comp Anim*, pp. 247–255, 2009.

[36] F. de Goes, C. Wallez, J. Huang, D. Pavlov, and M. Desbrun, "Power particles: an incompressible fluid solver based on power diagrams," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 50–1, 2015.

[37] F. Da, C. Batty, and E. Grinspun, "Multimaterial mesh-based surface tracking," *ACM Trans Graph*, vol. 33, no. 4, pp. 112:1–112:11, 2014.

[38] M. Sussman and M. Ohta, "A stable and efficient method for treating surface tension in incompressible two-phase flow," *SIAM J Sci Comp*, vol. 31, no. 4, pp. 2447–2471, 2009.

[39] I. Eckstein, J. Pons, Y. Tong, C. Kuo, and M. Desbrun, "Generalized surface flows for mesh processing," in *Proc Eurograph Symp Geom Proc*, pp. 183–192, Eurographics Association, 2007.

[40] M. Misztal, K. Erleben, A. Bargteil, J. Fursund, B. Christensen, J. Bærentzen, and R. Bridson, "Multiphase flow of immiscible fluids on unstructured moving meshes," *IEEE Trans Vis Comp Graph*, vol. 20, no. 1, pp. 4–16, 2013.

[41] M. Misztal and J. Bærentzen, "Topology-adaptive interface tracking using the deformable simplicial complex," *ACM Trans Graph (TOG)*, vol. 31, no. 3, pp. 1–12, 2012.

[42] M. Wicke, D. Ritchie, B. Klingner, S. Burke, J. Shewchuk, and J. O'Brien, "Dynamic local remeshing for elastoplastic simulation," *ACM Trans Graph*, vol. 29, no. 4, pp. 49:1–11, 2010.

[43] J. Brackbill, D. Kothe, and C. Zemach, "A continuum method for modeling surface tension," *J Comp Phys*, vol. 100, no. 2, pp. 335–354, 1992.

[44] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proc 2003 ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 154–159, Eurographics Association, 2003.

[45] J. Morris, "Simulating surface tension with smoothed particle hydrodynamics," *Int J Num Meth Fl*, vol. 33, no. 3, pp. 333–353, 2000.

[46] N. Akinci, G. Akinci, and M. Teschner, "Versatile surface tension and adhesion for sph fluids," *ACM Trans Graph (TOG)*, vol. 32, no. 6, pp. 1–8, 2013.

[47] M. Becker and M. Teschner, "Weakly compressible sph for free surface flows," in *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 209–217, 2007.

[48] S. Clavet, P. Beaudoin, and P. Poulin, "Particle-based viscoelastic fluid simulation," in *Proc 2005 ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 219–228, 2005.

[49] J. Yu, C. Wojtan, G. Turk, and C. Yap, "Explicit mesh surfaces for particle based fluids," *Comp Graph Forum*, vol. 31, no. 2pt4, pp. 815–824, 2012.

[50] M. Sandim, D. Cedrim, L. Nonato, P. Pagliosa, and A. Paiva, "Boundary detection in particle-based fluids," *Comp Graph For*, vol. 35, no. 2, pp. 215–224, 2016.

[51] Y. Zhang, B. Solenthaler, and R. Pajarola, "Adaptive sampling and rendering of fluids on the gpu," in *Proc Symp Point-Based Graph*, pp. 137–146, August 2008.

[52] G. Dilts, "Moving least-squares particle hydrodynamics ii: conservation and boundaries," *Int J Num Meth Eng*, vol. 48, no. 10, pp. 1503–1524, 2000.

[53] A. Haque and G. Dilts, "Three-dimensional boundary detection for particle methods," *J Comp Phys*, vol. 226, no. 2, pp. 1710–1730, 2007.

[54] X. He, N. Liu, G. Wang, F. Zhang, S. Li, S. Shao, and H. Wang, "Staggered meshless solid-fluid coupling," *ACM Trans Graph (TOG)*, vol. 31, no. 6, pp. 1–12, 2012.

[55] F. Zorilla, M. Ritter, J. Sappl, W. Rauch, and M. Harders, "Accelerating surface tension calculation in sph via particle classification and monte carlo integration," *Computers*, vol. 9, no. 2, p. 23, 2020.

[56] J. Orthmann, H. Hochstetter, B. J. J. Bader, S. Bayraktar, and A. Kolb, "Consistent surface model for sph-based fluid transport," in *Proc 12th ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 95–103, 2013.

[57] S. Popinet, "Numerical models of surface tension," *Annual Rev Fluid Mech*, vol. 50, pp. 49–75, 2018.

[58] E. Bänsch, "Finite element discretization of the navier–stokes equations with a free capillary surface," *Num Math*, vol. 88, no. 2, pp. 203–235, 2001.

[59] S. Hysing, "A new implicit surface tension implementation for interfacial flows," *Int J Num Meth Fl*, vol. 51, no. 6, pp. 659–672, 2006.

[60] J. Hochstein and T. Williams, "An implicit surface tension model," in *34th Aerospace Sciences Meeting and Exhibit*, p. 599, 1996.

[61] T. Hou, J. Lowengrub, and M. Shelley, "Removing the stiffness from interfacial flows with surface tension," *J Comp Phys*, vol. 114, no. 2, pp. 312–338, 1994.

[62] A. Jarauta, P. Ryzhakov, J. Pons-Prats, and M. Secanell, "An implicit surface tension model for the analysis of droplet dynamics," *J Comp Phys*, vol. 374, pp. 1196–1218, 2018.

[63] C. Schroeder, W. Zheng, and R. Fedkiw, "Semi-implicit surface tension formulation with a lagrangian surface mesh on an eulerian simulation grid," *J Comp Phys*, vol. 231, no. 4, pp. 2092–2115, 2012.

[64] A. Adamson and A. Gast, *Physical chemistry of surfaces*, vol. 150. Interscience Publishers New York, 1967.

[65] D. Sulsky, Z. Chen, and H. Schreyer, "A particle method for history-dependent materials," *Comp Meth App Mech Eng*, vol. 118, no. 1, pp. 179–196, 1994.

[66] C. Jiang, C. Schroeder, J. Teran, A. Stomakhin, and A. Selle, "The material point method for simulating continuum materials," in *ACM SIGGRAPH 2016 Course*, pp. 24:1–24:52, 2016.

[67] D. Hyde, S. Gagniere, A. Marquez-Razon, and J. Teran, "An implicit updated lagrangian formulation for liquids with large surface energy," *ACM Trans Graph*, vol. 39, Nov. 2020.

[68] E. Chernyaev, "Marching cubes 33: Construction of topologically correct isosurfaces," tech. rep., 1995.

[69] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Trans Vis Comp Graph*, vol. 18, no. 6, pp. 914–924, 2012.

[70] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Trans. Graph.*, vol. 21, p. 807–832, Oct. 2002.

[71] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A material point method for snow simulation," *ACM Trans Graph*, vol. 32, no. 4, pp. 102:1–102:10, 2013.

[72] D. Ram, T. Gast, C. Jiang, C. Schroeder, A. Stomakhin, J. Teran, and P. Kavehpour, "A material point method for viscoelastic fluids, foams and sponges," in *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 157–163, 2015.

[73] J. Chen, V. Kala, A. Marquez-Razon, E. Gueidon, D. Hyde, and J. Teran, "Supplementary technical document," tech. rep., 2021.

[74] L. Boyd and R. Bridson, "Multiflip for energetic two-phase fluid simulation," *ACM Trans Graph*, vol. 31, no. 2, pp. 16:1–16:12, 2012.

[75] D. Hyde, S. Gagniere, A. Marquez-Razon, and J. Teran, "Supplementary technical document," tech. rep., 2020.

[76] B. Smith, F. Goes, and T. Kim, "Analytic eigensystems for isotropic distortion energies," *ACM Trans Graph (TOG)*, vol. 38, no. 1, pp. 1–15, 2019.

[77] T. Kim, F. D. Goes, and H. Iben, "Anisotropic elasticity for inversion-safety and element rehabilitation," *ACM Trans Graph (TOG)*, vol. 38, no. 4, pp. 1–15, 2019.

[78] J. Teran, E. Sifakis, G. Irving, and R. Fedkiw, "Robust quasistatic finite elements and flesh simulation," in *Proc 2005 ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 181–190, 2005.

[79] R. Rioboo, C. Tropea, and M. Marengo, "Outcomes from a drop impact on solid surfaces," *Atomization and Sprays*, vol. 11, no. 2, 2001.

[80] T. Young, "III. an essay on the cohesion of fluids," *Phil Trans Royal Soc London*, vol. 95, pp. 65–87, 1805.

[81] P. Clausen, M. Wicke, J. R. Shewchuk, and J. F. O'Brien, "Simulating liquids and solid-liquid interactions with lagrangian meshes," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 2, p. 17, 2013.

[82] M. Ding, X. Han, S. Wang, T. Gast, and J. Teran, "A thermomechanical material point method for baking and cooking," *ACM Trans Graph*, vol. 38, no. 6, p. 192, 2019.

[83] C. Gissler, A. Henne, S. Band, A. Peer, and M. Teschner, "An implicit compressible sph solver for snow simulation," *ACM Trans Graph (TOG)*, vol. 39, no. 4, pp. 36–1, 2020.

[84] A. Chorin, "A numerical method for solving incompressible viscous flow problems," *J Comp Phys*, vol. 2, no. 1, pp. 12–26, 1967.

[85] F. Harlow and E. Welch, "Numerical calculation of time dependent viscous flow of fluid with a free surface," *Phys Fluid*, vol. 8, no. 12, pp. 2182–2189, 1965.

[86] J. Stam, "Stable fluids," in *Siggraph*, vol. 99, pp. 121–128, 1999.

[87] A. Robert, "A stable numerical integration scheme for the primitive meteorological equations," *Atm Ocean*, vol. 19, no. 1, pp. 35–46, 1981.

[88] F. Harlow, "The particle-in-cell method for numerical solution of problems in fluid dynamics," *Meth Comp Phys*, vol. 3, pp. 319–343, 1964.

[89] J. Brackbill and H. Ruppel, "FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions," *J Comp Phys*, vol. 65, pp. 314–343, 1986.

[90] Y. Zhu and R. Bridson, "Animating sand as a fluid," *ACM Trans Graph*, vol. 24, no. 3, pp. 965–972, 2005.

[91] A. Stomakhin, C. Schroeder, C. Jiang, L. Chai, J. Teran, and A. Selle, "Augmented MPM for phase-change and varied materials," *ACM Trans Graph*, vol. 33, no. 4, pp. 138:1–138:11, 2014.

[92] R. Bridson, *Fluid simulation for computer graphics*. Taylor & Francis, 2008.

[93] T. Hughes, *The finite element method : linear static and dynamic finite elment analysis*. Mineola, NY : Dover Publications, 2000.

[94] C. Taylor and P. Hood, "A numerical solution of the navier-stokes equations using the finite element technique," *Comp & Fl*, vol. 1, no. 1, pp. 73–100, 1973.

[95] C. de Boor, *A Practical Guide to Splines*. Springer, 1978.

[96] A. Bressan, "Isogeometric regular discretization for the stokes problem," *IMA J Num Anal*, vol. 31, no. 4, pp. 1334–1356, 2010.

[97] M. Steffen, R. Kirby, and M. Berzins, "Analysis and reduction of quadrature errors in the material point method (MPM)," *Int J Numer Meth Eng*, vol. 76, no. 6, pp. 922–948, 2008.

[98] L. Evans, *Partial differential equations*. Providence, R.I.: American Mathematical Society, 2010.

[99] J. Sawyer, "A semi-lagrangian method of solving the vorticity advection equation," *Tellus*, vol. 15, no. 4, pp. 336–342, 1963.

[100] B. Kim, Y. Liu, I. Llamas, and J. Rossignac, "Advections with significantly reduced dissipation and diffusion," *IEEE Trans Viz Comp Graph*, vol. 13, no. 1, pp. 135–144, 2006.

[101] B. Kim, Y. Liu, I. Llamas, and J. Rossignac, "Flowfixer: Using bfecc for fluid simulation," in *Proc Eurograph Conf Nat Phen*, pp. 51–56, Eurographics Association, 2005.

[102] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac, "An unconditionally stable maccormack method," *J Sci Comp*, vol. 35, no. 2-3, pp. 350–371, 2008.

[103] D. Xiu and G. Karniadakis, "A semi-lagrangian high-order method for navier–stokes equations," *J Comp Phys*, vol. 172, no. 2, pp. 658–684, 2001.

[104] C. Schroeder, A. Stomakhin, R. Howes, and J. Teran, "A second order virtual node algorithm for navier-stokes flow problems with interfacial forces and discontinuous material properties," *J Comp Phys*, vol. 265, pp. 221 – 245, 2014.

[105] N. Chentanez, M. Müller, and T. Kim, "Coupling 3d eulerian, heightfield and particle methods for interactive simulation of large scale liquid phenomena," *IEEE Trans Vis Comp Graph*, vol. 21, no. 10, pp. 1116–1128, 2015.

[106] F. Ferstl, R. Ando, C. Wojtan, R. Westermann, and N. Thuerey, "Narrow band flip for liquid simulations," in *Comp Graph For*, vol. 35, pp. 225–232, Wiley Online Library, 2016.

[107] T. Sato, C. Wojtan, N. Thuerey, T. Igarashi, and R. Ando, "Extended narrow band FLIP for liquid simulations," *Comp Graph For*, 2018.

[108] R. Fedkiw, J. Stam, and H. Jensen, "Visual simulation of smoke," in *SIGGRAPH*, pp. 15–22, ACM, 2001.

[109] J. Steinhoff and D. Underhill, "Modification of the euler equations for "vorticity confinement": Application to the computation of interacting vortex rings," *Phys Fl*, vol. 6, no. 8, pp. 2738–2744, 1994.

[110] D. Kim, O. Song, and H. Ko, "A semi-lagrangian cip fluid solver without dimensional splitting," in *Comp Graph For*, vol. 27, pp. 467–475, Wiley Online Library, 2008.

[111] T. Yabe, F. Xiao, and T. Utsumi, "The constrained interpolation profile method for multiphase analysis," *J Comp Phys*, vol. 169, pp. 556–593, 2001.

[112] O. Song, D. Kim, and H. Ko, "Derivative particles for simulating detailed movements of fluids," *IEEE Trans Vis Comp Graph*, pp. 247–255, 2009.

[113] J. Molemaker, J. Cohen, S. Patel, and J. Noh, "Low viscosity flow simulations for animation," in *Proc 2008 ACM SIGGRAPH/Eurographics Symp Comp Anim*, pp. 9–18, Eurographics Association, 2008.

[114] B. Leonard, "A stable and accurate convective modelling procedure based on quadratic upstream interpolation," *Computer methods in applied mechanics and engineering*, vol. 19, no. 1, pp. 59–98, 1979.

[115] R. Courant, E. Isaacson, and M. Rees, "On the solution of nonlinear hyperbolic differential equations by finite differences," *Comm Pure App Math*, vol. 5, no. 3, pp. 243–255, 1952.

[116] A. Staniforth and J. Côté, "Semi-lagrangian integration schemes for atmospheric models-a review," *Monthly weather review*, vol. 119, no. 9, pp. 2206–2223, 1991.

[117] M. Falcone and R. Ferretti, "Convergence analysis for a class of high-order semi-lagrangian advection schemes," *SIAM J Num Anal*, vol. 35, no. 3, pp. 909–940, 1998.

[118] J. Pudykiewicz and A. Staniforth, "Some properties and comparative performance of the semi-lagrangian method of robert in the solution of the advection-diffusion equation," *Atmosphere-Ocean*, vol. 22, no. 3, pp. 283–308, 1984.

[119] J. Bates and A. McDonald, "Multiply-upstream, semi-lagrangian advective schemes: Analysis and application to a multi-level primitive equation model," *Monthly Weather Review*, vol. 110, no. 12, pp. 1831–1842, 1982.

[120] F. Cheng, X. Wang, and B. Barsky, "Quadratic b-spline curve interpolation," *Comp Math App*, vol. 41, no. 1-2, pp. 39–50, 2001.

[121] W. Cheney and D. Kincaid, *Numerical mathematics and computing*. Cengage Learning, 2012.

[122] P. Makar and S. Karpik, "Basis-spline interpolation on the sphere: Applications to semi-lagrangian advection," *Monthly Weather Review*, vol. 124, no. 1, pp. 182–199, 1996.

[123] L. Riishøjgaard, S. Cohn, Y. Li, and R. Ménard, "The use of spline interpolation in semi-lagrangian transport models," *Monthly Weather Review*, vol. 126, no. 7, pp. 2008–2016, 1998.

[124] J. Wang and A. Layton, "New numerical methods for burgers' equation based on semi-lagrangian and modified equation approaches," *App Num Math*, vol. 60, no. 6, pp. 645–657, 2010.

[125] P. Mullen, K. Crane, D. Pavlov, Y. Tong, and M. Desbrun, "Energy-preserving integrators for fluid animation," in *ACM Trans Graph (TOG)*, vol. 28, p. 38, ACM, 2009.

[126] Z. Qu, X. Zhang, M. Gao, C. Jiang, and B. Chen, "Efficient and conservative fluids using bidirectional mapping," *ACM Trans. Graph.*, vol. 38, no. 4, 2019.

[127] J. Tessendorf and B. Pelfrey, "The characteristic map for fast and efficient vfx fluid simulations," 2011.

[128] T. Sato, T. Igarashi, C. Batty, and R. Ando, "A long-term semi-lagrangian method for accurate velocity advection," in *SIGGRAPH Asia 2017 Tech Briefs*, p. 5, ACM, 2017.

[129] T. Sato, C. Batty, T. Igarashi, and R. Ando, "Spatially adaptive long-term semi-lagrangian method for accurate velocity advection," *Comp Vis Med*, vol. 4, no. 3, pp. 223–230, 2018.

[130] A. Selle, N. Rasmussen, and R. Fedkiw, "A vortex particle method for smoke, water and explosions," in *ACM Trans Graph (TOG)*, vol. 24, pp. 910–914, ACM, 2005.

[131] A. Angelidis and F. Neyret, "Simulation of smoke based on vortex filament primitives," in *Proc 2005 ACM SIGGRAPH/Eurographics Symp Comp Anim*, pp. 87–96, ACM, 2005.

[132] A. Chern, F. Knöppel, U. Pinkall, P. Schröder, and S. Weissmann, "Schrödinger's smoke," *ACM Trans Graph (TOG)*, vol. 35, no. 4, p. 77, 2016.

[133] E. Sharif, Y. Tong, E. Kanso, P. Schröder, and M. Desbrun, "Stable, circulation-preserving, simplicial fluids," *ACM Trans Graph (TOG)*, vol. 26, no. 1, p. 4, 2007.

[134] S. Park and M. Kim, "Vortex fluid for gaseous phenomena," in *Proc 2005 ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 261–270, ACM, 2005.

[135] S. Weissmann and U. Pinkall, "Filament-based smoke with vortex shedding and variational reconnection," in *ACM Trans Graph (TOG)*, vol. 29, p. 115, 2010.

[136] J. Zehnder, R. Narain, and B. Thomaszewski, "An advection-reflection solver for detail-preserving fluid simulation," *ACM Trans Graph (TOG)*, vol. 37, no. 4, p. 85, 2018.

[137] R. Narain, J. Zehnder, and B. Thomaszewski, "A second-order advection-reflection solver," *Proc ACM Comput Graph Interact Tech*, vol. 2, July 2019.

[138] X. Zhang, R. Bridson, and C. Greif, "Restoring the missing vorticity in advection-projection fluid solvers," *ACM Trans Graph (TOG)*, vol. 34, no. 4, p. 52, 2015.

[139] D. Enright, D. Nguyen, F. Gibou, and R. Fedkiw, "Using the particle level set method and a second order accurate pressure boundary condition for free surface flows," in *ASME/JSME 2003 4th Joint Fluids Summer Engineering Conference*, pp. 337–342, ASMEDC, 2003.

[140] C. Batty, F. Bertails, and R. Bridson, "A fast variational framework for accurate solid-fluid coupling," *ACM Trans Graph*, vol. 26, no. 3, 2007.

[141] C. Batty and R. Bridson, "Accurate viscous free surfaces for buckling, coiling, and rotating liquids," *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 219–228, 2008.

[142] E. Larionov, C. Batty, and R. Bridson, "Variational stokes: A unified pressure-viscosity solver for accurate viscous liquids," *ACM Trans Graph*, vol. 36, no. 4, 2017.

[143] T. Belytschko, R. Gracie, and G. Ventura, "A review of extended/generalized finite element methods for material modeling," *Mod Sim Mat Sci Eng*, vol. 17, no. 4, p. 043001, 2009.

[144] D. Koschier, J. Bender, and N. Thuerey, "Robust extended finite elements for complex cutting of deformables," *ACM Trans Graph*, vol. 36, no. 4, pp. 55:1–55:13, 2017.

[145] T. Hughes, J. Cottrell, and Y. Bazilevs, "Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement," *Comp Meth App Mech Eng*, vol. 194, no. 39,41, pp. 4135–4195, 2005.

[146] T. Rüberg and F. Cirak, "Subdivision-stabilised immersed b-spline finite elements for moving boundary flows," *Comp Meth App Mech Eng*, vol. 209, pp. 266–283, 2012.

[147] Y. Bazilevs and T. Hughes, "Weak imposition of dirichlet boundary conditions in fluid mechanics," *Comp Fluids*, vol. 36, no. 1, pp. 12–26, 2007.

[148] E. Kim, "A mixed galerkin method for computing the flow between eccentric rotating cylinders," *Int J Num Meth Fl*, vol. 26, no. 8, pp. 877–885, 1998.

[149] A. Kravchenko, P. Moin, and K. Shariff, "B-spline method and zonal grids for simulations of complex turbulent flows," *J Comp Phys*, vol. 151, no. 2, pp. 757–789, 1999.

[150] O. Botella, "On a collocation b-spline method for the solution of the navier–stokes equations," *Comp Fl*, vol. 31, no. 4-7, pp. 397–420, 2002.

[151] R. Ando, N. Thürey, and C. Wojtan, "Highly adaptive liquid simulations on tetrahedral meshes," *ACM Trans Graph*, vol. 32, no. 4, pp. 103:1–103:10, 2013.

[152] H. Kuo and R. Williams, "Semi-lagrangian solutions to the inviscid burgers equation," *Monthly Weather Review*, vol. 118, no. 6, pp. 1278–1288, 1990.

[153] S. Gagniere, D. Hyde, A. Marquez-Razon, C. Jiang, Z. Ge, X. Han, Q. Guo, and J. Teran, "Supplementary technical document," tech. rep., 2020.

[154] D. Demidov, "Amgcl: An efficient, flexible, and extensible algebraic multigrid implementation," *Lobachevskii Journal of Mathematics*, vol. 40, pp. 535–546, 5 2019.

[155] Y. Saad, *Iterative Methods for Sparse Linear Systems*. USA: Society for Industrial and Applied Mathematics, 2nd ed., 2003.

[156] C. Shah, D. Hyde, H. Qu, and P. Levis, "Distributing and load balancing sparse fluid simulations," *Computer Graphics Forum*, vol. 37, no. 8, pp. 35–46, 2018.

[157] F. Zhang, X. Zhang, Y. Sze, Y. Lian, and Y. Liu, "Incompressible material point method for free surface flow," *J Comp Phys*, vol. 330, no. C, pp. 92–110, 2017.

[158] C. Huang, "Semi-lagrangian advection schemes and eulerian wkl algorithms," *Monthly Weather Review*, vol. 122, no. 7, pp. 1647–1658, 1994.