UNIVERSITY OF CALIFORNIA

Los Angeles

The Material Point Method
for Solid and Fluid Simulation

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

by

Qi Guo

2020

ABSTRACT OF THE DISSERTATION

The Material Point Method for Solid and Fluid Simulation

by

Qi Guo
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2020
Professor Joseph M. Teran, Chair

The Material Point Method (MPM) has shown its high potential for physics-based simulation in the area of computer graphics. In this dissertation, we introduce a couple of improvements to the traditional MPM for different applications and demonstrate the advantages of our methods over the previous methods.

First, we present a generalized transfer scheme for the hybrid Eulerian/Lagrangian method: the Polynomial Particle-In-Cell Method (PolyPIC). PolyPIC improves kinetic energy conservation during transfers, which leads to better vorticity resolution in fluid simulations and less numerical damping in elastoplasticity simulations. Our transfers are designed to select particle-wise polynomial approximations to the grid velocity that are optimal in the local mass-weighted L^2 norm. Indeed our notion of transfers reproduces the original Particle-In-Cell Method (PIC) and recent Affine Particle-In-Cell Method (APIC). Furthermore, we derive a polynomial basis that is mass-orthogonal to facilitate the rapid solution of the optimality condition. Our method applies to both of the collocated and staggered grid.

As the second contribution, we present a novel method for the simulation of thin shells with frictional contact using a combination of MPM and subdivision finite elements. The shell kinematics are assumed to follow a continuum shell model which is decomposed into a Kirchhoff-Love motion that rotates the mid-surface normals followed by shearing and compression/extension of the material along the mid-surface normal. We use this decomposition to design an elastoplastic constitutive model to resolve frictional contact by decoupling resistance to contact and shearing from the bending resistance components of stress. We show that by resolving frictional contact with a continuum approach, our hybrid Lagrangian/Eulerian approach is capable of simulating challenging shell contact scenarios with hundreds of thousands to millions of degrees of freedom. Without the need for collision detection or resolution, our method runs in a few minutes per frame in these high-resolution examples. Furthermore, we show that our technique naturally couples with other traditional MPM methods for simulating granular and related materials.

In the third part, we present a new hybrid Lagrangian Material Point Method for simulating volumetric objects with frictional contact. The resolution of frictional contact in the thin shell simulation cannot be generalized to the case of volumetric materials directly. Also, even though MPM allows for the natural simulation of hyperelastic materials represented with Lagrangian meshes, it usually coarsens the degrees of freedom of the Lagrangian mesh and can lead to artifacts, e.g., numerical cohesion. We demonstrate that our hybrid method can efficiently resolve these issues. We show the efficacy of our technique with examples that involve elastic soft tissues coupled with kinematic skeletons, extreme deformation, and coupling with various elastoplastic materials. Our approach also naturally allows for two-way rigid body coupling.

The dissertation of Qi Guo is approved.

Jeffrey D. Eldredge

Luminita A. Vese

Christopher R. Anderson

Joseph M. Teran, Committee Chair

University of California, Los Angeles

2020

To my grandma, KOU Miao-kun

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my highest gratitude to my advisor, Professor Joseph Teran, for his continuous guidance throughout my Ph.D. career. His insights, expertise, and broad knowledge of this research area are invaluable to me and help me grow as a researcher. I also thank him for providing me many great opportunities.

I would also like to thank Professor Jeff Eldredge, Professor Luminita Vese, and Professor Chris Anderson for serving on my committee and providing further advice on my study and work.

I want to thank Chenfanfu Jiang and Ted Gast for their guidance when I started my research career in the group. I thank my colleagues, Xuchen Han and Mengyuan Ding, for their tremendous help. I also want to thank all my collaborators.

My appreciation goes to Zhili Chen, Qi Sun, Joel Wretborn, Greg Klár, Ken Museth, Beibei Liu, and Yuting Ye for sharing their knowledge and help during my internships at Adobe Research, Weta Digital, and Facebook Reality Labs.

I want to thank my girlfriend, Chuxin Donna Wei, for her love and unconditional support. She encourages me a lot and always alleviates my anxiety.

My sincere gratitude goes to all my friends, and I could not have these wonderful and incredible years without their company. These golden days with my friends will never be forgotten.

Finally, I thank my parents and my grandparents for everything they did. Love you all.

VITA

2015–2020	PhD Student, University of California - Los Angeles, Los Angeles, California, U.S.
2019	Research Intern, Facebook Reality Labs
2019	Simulation Intern, Weta Digital
2018	Research Intern, Adobe Research
2013–2015	M.S., University of Washington, Seattle, Washington, U.S.
2009–2013	B.S., Shandong University, Jinan, Shandong, China

PUBLICATIONS

X. Han, T. Gast, Q. Guo, S. Wang, C. Jiang, J. Teran, "A Hybrid Material Point Method for Frictional Contact with Diverse Materials", *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2, 2, 1-24, 2019.

Q. Guo, X. Han, C. Fu, T. Gast, R. Tamstorf, J. Teran, "A Material Point Method for Thin Shells with Frictional Contact", *ACM Trans Graph* 37, 4, 147, 2018.

M. Gao, A. Pradhana, X. Han, Q. Guo, G. Kot, E. Sifakis, and C. Jiang. "Animating Fluid Sediment Mixture in Particle-laden Flows". *ACM Trans Graph* 37, 4, 149:1–149:11, 2018.

B. Deconinck, Q. Guo, E. Shlizerman, V. Vasan. "Fokas's uniform transform method for linear systems", *Quart Appl Math* 76, 463-488, 2018.

C. Fu, Q. Guo, T. Gast, C. Jiang, and J. Teran. “A Polynomial Particle-in-cell Method”. *ACM Trans Graph* 36, 6, 222:1–222:12, 2017.

CHAPTER 1

Introduction

Simulating natural phenomena remains hugely challenging these days. The Material Point Method (MPM) rises as the generalization of Particle-In-Cell Method (PIC) and Fluid Implicit Particle Method (FLIP) to solid mechanics [Har64, BR86, SZS95], has been shown to be a very effective hybrid Eulerian/Lagrangian method for simulating various materials.

In this dissertation, we raise a computationally efficient algorithm, the Polynomial Particle-In-Cell Method, for the hybrid Eulerian/Lagrangian which enhances the kinetic energy conservation. Also, we develop a new method using MPM for the numerical simulation of the continuum-based shell model. Lastly, we design a novel hybrid Lagrangian MPM to alleviate a couple of drawbacks of collision handling when simulating volumetric objects with the traditional MPM.

1.1 Material Point Method

MPM combines Lagrangian material particles with Eulerian Cartesian grids, and it handles the phenomena like fracture/topological change, multiple material interactions, and challenging self contact scenarios with complex geometric domains very well. This was first demonstrated for snow dynamics by Stomakhin et al. [SSC13]. Since then a wide variety of other phenomena, particularly those that can be described as elastoplastic, have been simulated with MPM in graphics applications. This includes the dynamics of non-Newtonian fluids and foams [YSB15, RGJ15], melting [SSJ14, GTJ17, DHW19], porous media [TGK17, GPH18, FBG18], and frictional contact between granular materials [DB16, KGP16, YSC18]. MPM has also been used to simulate contact and collision with volumetric elastic objects

[JSS15, ZZL17] and frictional contact between thin hyperelastic materials like clothing and hair [JGT17, FBG18].

We summarize the MPM steps as below:

1. **Particles to grid**: Transfer particle mass and momentum to the grid.
2. **Apply forces**: Compute the elastic force, and the external force on the grid and update grid velocities based on these forces.
3. **Grid to particles**: Interpolate velocities from the grid to update particles' velocities.
4. **Strain and representation update**: Update the positions, deformation gradient and the trial elastic state.
5. **Plasticity update**: Update the plastic deformation gradient using the plastic flow return mapping algorithm.

1.2 Polynomial Particle-In-Cell Method

The Affine Particle-In-Cell (APIC) Method was proposed by Jiang et al.[JSS15, JST17] to improve the accuracy of the transfers in the hybrid Lagrangian/Eulerian method by augmenting each particle with a locally affine, rather than locally constant description of the velocity. This reduced the dissipation of the original method without suffering from the noise. In this dissertation, we present a generalization of APIC, Polynomial Particle-In-Cell method (PolyPIC), by augmenting each particle with a more general local function. By viewing the grid-to-particle transfer as a linear and angular momentum conserving projection of the particle-wise local grid velocities onto a reduced basis, we greatly improve the energy and vorticity conservation over the original APIC. Furthermore, we show that the cost of PolyPIC is negligible over APIC when using a particular class of local polynomial functions. Lastly, we note that our method retains the filtering property of APIC and PIC and thus has similar robustness to noise.

We summarize our contributions as:

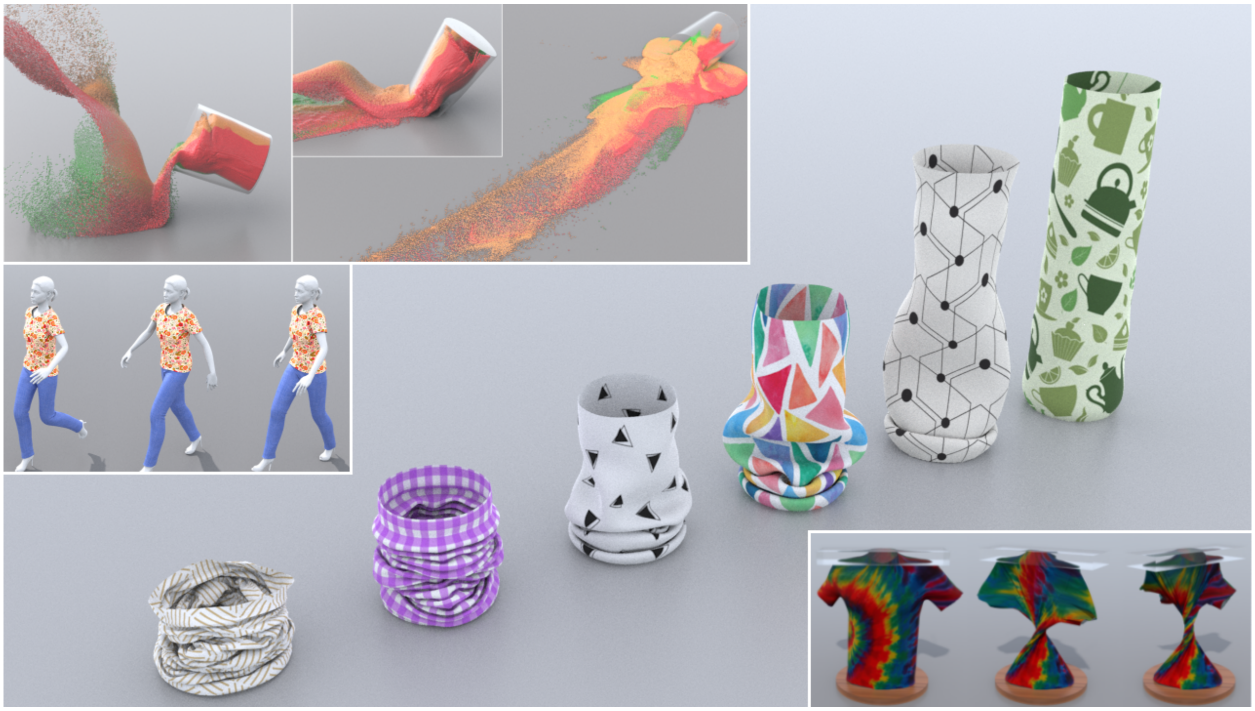

Figure 1.1: **Shell Montage**. Upper left : simulation of shells coupled with granular materials. Center left : a walk cycle benchmark for clothing simulation. Bottom right : a T-shirt twisted to induce many self-collisions. Center : the effect of increasing bending stiffness (from left to right) for six collapsing elastic cylinders.

- A generalization of APIC from locally affine to locally polynomial representations that improves kinetic energy conservation in particle/grid transfers.
- A mass weighted L^2 optimality condition that achieves linear and angular momentum conservation.
- A mass-orthogonal class of polynomials for rapid solution of projection to the polynomial basis.
- Natural treatment of staggered and collocated grids.

We demonstrate the benefits of our technique in a number of representative applications of incompressible flow and MPM simulation of elastoplastic materials.

1.3 Shell

The shell structure is a thin surface with stretching, shearing, and bending resistance. This structure appears almost everywhere in the real-world products made of sheets of different materials; see Figure 1.1. The numerical simulation of the thin shell draws a long history and is usually based on the finite shell element, which is less computationally expensive and more numerically stable than the volumetric solid finite element [BLM13].

For the shell model, two types of continuum-based assumptions, the Kirchhoff-Love theory, and Mindlin-Reissner theory, are widely adopted to model the elastic behavior. Additionally, in many cases, the permanent plastic deformation of the shell is another crucial factor to be counted. Furthermore, in the challenging large-scale simulation scenario, for example, in visual effects, the intense frictional contact/collision, together with the coupling effect with diverse materials, needs be captured during the simulation.

We present an algorithm that takes all the elements above into account. We demonstrate the efficacy of our approach with several challenging simulations for shell and clothing simulation applications with scenarios involving hundreds of thousands to millions of degrees of freedom. Without the need for collision detection or resolution, we show that our method runs in a few minutes per frame in these high-resolution examples. We summarize our novel contributions as

- An elastoplastic formulation for frictional contact and resistance to bending and denting of thin shells

- A strain splitting technique to separate thin shell motion into Kirchhoff-Love and continuum shell components

- A plane strain/stress formulation for Kirchhoff-Love thin shells that simplifies the return mapping algorithm for denting resistance

- A hybrid/Eulerian MPM discretization of the deformation gradient in the shell and the associated potential energy

1.4 Volumetric Objects with Frictional Contact

The traditional MPM update handles the collision automatically because particle movements are interpolated from undistortable nodal movement on a grid. However, there are several drawbacks of this treatment of collision. For volumetric objects, this scheme is unable to regulate the potential energy with a plasticity model derived from Coulomb friction as for the thin shell simulation in Section §1.3. Additionally, the method requires careful matching between grid and mesh resolution. In the situations where the resolutions mismatch, the traditional MPM either does not prevent collision at all, or causes visual artifacts, such as interaction at a visual distance. Furthermore, when volumetric objects are coupled with traditional MPM materials, for example, the granular materials, it causes numerical cohesion of materials, see Figure 1.2.

In this dissertation, we develop a novel hybrid Lagrangian Material Point Method to alleviate these drawbacks. Our approach utilizes more of the Lagrangian degrees of freedom to define novel alternatives to the updated Lagrangian assumption. We retain aspects of MPM that allow for collision resolution without suffering from information loss when going from particles to grid. Our approach also resolves the Eulerian grid size (and apparent separation distance) limitations associated with volumetric elasticity, allowing for Coulomb frictional contact with volumetric elastic meshes. We support coupling with materials simulated with standard MPM discretizations, and we provide for simple two-way coupling with rigid bodies. We demonstrate the effectiveness of our techniques with skinning, clothing, and multi-material simulation examples. In summary, our contributions are:

- Collision impulses defined from the MPM particle to grid transfers that are applied to Lagrangian FEM volumetric meshes for frictional self collision, removing the drawbacks of the volumetric approaches in [JSS15, ZZL17].
- Two-way coupling with rigid bodies.
- Removal of numerical cohesion between phases.
- Coupling with materials discretized with traditional MPM.

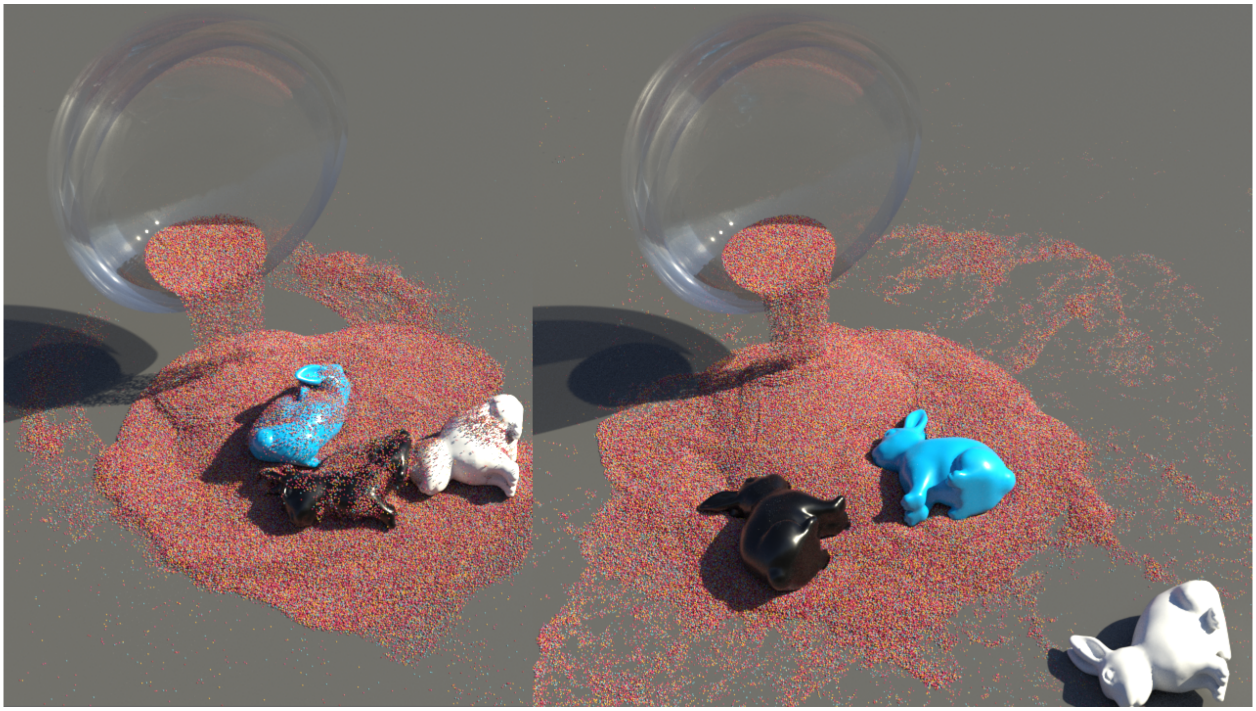

Figure 1.2: **Friction**. Our method (right) removes the excessive numerical friction common to traditional MPM (left), and regulates friction with the Coulomb friction model. With low friction coefficients, the colored sand freely slides off the bunnies.

1.5 Related Work

1.5.1 Particle-In-Cell Method

Momentum conservation and noise removal: There are a number of recent PIC approaches designed to improve robustness to noise without sacrificing accurate energy and momentum conservation. Hammerquist and Nairn [HN17] developed a PIC extension designed to reduce the noise of the FLIP by adding a smoothing term to the FLIP velocity. This strikes a good balance between noise reduction and energy preservation. Edwards and Bridson also add a regularization term to diminish particle noise [EB12]. Gritton and Berzins [GB17] reduce noise by filtering spatial gradients based on a local SVD approximation of the null space of the particle-to-grid transfer operator. Wallstedt and Guilkey use a locally-affine assumption as in [JSS15, JST17], but they use FLIP grid-to-particle transfers that still suffer from noise [WG07]. Um et al. develop a particle repulsion force to improve particle bunching associated with the ringing instability [UBH14].

1.5.2 Shell

The literature in graphics and engineering related to simulation of clothing is extremely vast. Here we only discuss the work most related to our continuum-based shell model.

Early continuum models: Continuum models for elastic surfaces with appreciable bending resistance have been used for many years. Early methods had many limitations related to treatment of self-collision, general mesh geometry/topology etc., but they demonstrated great promise and addressed aspects of the functionality we provide with our method. As an example, Terzopoulos et al.[TPB87] use the second fundamental form to define a bending energy and use finite differences to discretize the problem over a regular grid. Eischen et al.[EDC96] use a finite element method (FEM) discretization of Mindlin-Reissner shells to model quasistatic equilibrium with draping cloth. They use penalty methods for collision with external objects, but they do not handle self-collision. Other early works use continuum shell models like those of Simo et al.[SF89] successfully for fabrics, albeit with limited support for self-collision [CCO91, GLS95, CG95, MS07]. Etzmuss et al.[EKS03, EGS03] go further by approximating the bending response in cloth warp and weft directions using a discrete projected Laplacian, but the Laplacian approach is limited to flat reference configurations.

Kirchhoff-Love Theory: The Kirchhoff-Love model is one example of a continuum model used for thin shells. The formulation applies the simple kinematic assumption that lines normal to the shell mid-surface always remain normal as the shell is deformed. However, the kinematic assumption requires higher order derivatives in the associated PDEs and this requires comparatively burdensome regularity of interpolation functions used in FEM calculations. Many approaches in engineering and graphics applications use Kirchhoff-Love continuum shells despite the additional regularity requirements. However, very few of them address the problem of self and external object collision. Cirak et al.[COS00] use Loop's subdivision scheme for triangle meshes to develop FEM basis functions that are H^2 as required by Kirchhoff-Love theory for thin shells. Similarly, Wawrzinek et al.[WHP11] use

the Catmull-Clark subdivision scheme and Lu and Zheng [LZ14] use the NURBS isogeometric analysis discretization of Kirchhoff-Love shells in Kiendl et al.[KBL09, KHW15]. Of these, only Lu and Zheng [LZ14] address self and external body collisions, and Wawrzinek et al.[WHP11] and Cirak et al.[COS00] only focus on linear elasticity and small strain problems. Cirak and Ortiz[CO01] provide an extension of the subdivision approach in [COS00] to allow for large deformation and nonlinear elasticity. A number of works including Thomaszewski et al.[TWS06], Grinspun et al.[GCS99] and Kaufmann et al.[KMB09] provide extensions to include inertia effects, implicit time stepping and large deformations. Kaufmann et al.[KMB09] use the Discontinous Galerkin (DG) approach of [NR08] to remove the need for H^2 interpolation. However, the DG approach requires the duplication of grid nodes on element boundaries which increases computational expense. Martin et al.[MKB10] use Kirchhoff-Love shell and Kirchhoff rod models as motivation in their construction of a unified approach to simulation of elastic volumes, solids and rods. Clyde et al.[CTT17] design a new orthotropic hyperelastic constitutive model for Kirchhoff-Love shells simulated with subd FEM to fit experimental data, but they also do not provide a treatment for self-collision. Grinspun et al.[GKS02] develop an adaptive version of the subd interpolation functions. Similar to our approach, Long et al. [LBC12] show that shear-flexible shells can be decomposed into Kirchhoff-Love and shear motions. Remarkably they show that the splitting has no compatibility constraints on the shape functions used for discretizing the mid-surface and the shear vectors respectively. Echter et al. [EOB13] use a family of isogeometric shell finite elements based on NURBS shape functions to satisfy the H^2 regularity requirements. Furthermore, as with our approach they split the shell kinematics into bending and shear deformations and show that this results in an element that prevents shear locking for Mindlin-Reissner shell kinematics.
Plasticity for wrinkles: Our approach naturally supports plasticity based denting of shells. Similar approaches in the literature include that of Gingold et al.[GSH04] where they use a von-Mises yield condition with kinematic hardening to create denting and wrinkling effects. Narain et al.[NPO13] develop an adaptive method for triangle meshes to simulate detailed folds and wrinkles. They use the hinge bending models in Grinspun et al.[GHD03] and Bridson et al.[BMF03] with nonzero rest angles derived from the plasticity formulation in

Gingold et al.[GSH04]. Our plasticity approach for producing wrinkling behavior is very similar to these. However because we also use plasticity to model frictional contact, we design a decoupled plasticity model.

1.5.3 Frictional Contact and Collision

Lagrangian/Eulerian collision/contact: A number of recent approaches have used hybrid Lagrangian and Eulerian views to simplify collision and contact treatment. Examples include simulation of elastoplastic solids using an Eulerian view of the governing physics [LLJ11, FLL13, LSN13, FLP14], using MPM [YSB15, SSC13, ZB05, NGL10, DB16, KGP16] and using Particle-In-Cell [MSW09]. Other hybrid approaches have been used successfully for solid/fluid coupling [TLK16, JSS15, JGT17] and for crowds [NGC09, GNL14]. Hybrid approaches of this type provide the starting point for our approach.

Shell contact and collision: Collision and contact handling for subd and NURBS based interpolation is challenging because of wider coupling between discrete degrees of freedom; however added regularity in the surface can simplify some aspects [MCB13, MB16]. A detailed review of contact with isogeometric approaches for volumetric objects are provided in Temizer et al.[TWH11] and Lorentis et al.[LWH14]. Martin et al.[MKB10] use forces derived from an energy that penalizes overlap of particles. Our collision stress response similarly arises from a potential, however their approach is purely Lagrangian whereas our is hybrid Lagrange/Eulerian. A number of works including Lu and Zheng [LZ14] and Thomaszewski et al.[TWS06] use the Bridson et al. impulse based approach for self collision. However, the Bridson et al. approach [BFA02] to self collision is designed for linear strain triangle meshes, this makes their application to more general meshes using subd and NURBS more challenging. Specifically, a triangulated mesh must be created solely for collision purposes and the application of the impulse can only be applied assuming linear interpolation, which is innacurate. Lu and Zheng [LZ14] use the NURBS isogeometric analysis discretization of Kirchhoff-Love shells in Kiendl et al.[KBL09, KHW15]. They use collision detection techniques from Lu[Lu11] and Temizer et al.[TWH11]. Grinspun et al.[GCS99] use a variational approach to self collision that is rooted in the approach of Kane et al.[KRO99] and the subdivision-surface interference detection algorithm in Grinspun and Schröder[GS01]. Kane et al. [KRO99] use non-smooth analysis to formulate self-collision in a Newmark (implicit/explicit) time stepping schemes as nonlinearly constrained optimization problems which they solve with sequential quadratic programming (SQP). However, the approach in Kane et al. [KRO99] is computationally burdensome for simulations with moderate to high spatial mesh resolution.

CHAPTER 2

Polynomial Particle-In-Cell Method

2.1 Method Outline and Notation

Our method is concerned with the update of the Lagrangian quantities in PIC calculations. We discuss this in detail and give an overview of each step in the process in Section §2.4. However, we first motivate our generalized notion of velocity local to a particle in Section §2.2 as well as the connection of our method to the very useful class of updated Lagrangian techniques in Section §2.3.

The Lagrangian state associated with particle p at time t^n consists of mass m_p, position $\mathbf{x}_p^n$, generalized velocity coefficients $\mathbf{c}_p^n$ and auxiliary quantities $\mathbf{A}_p^n$. Note that the mass does not change with time in accordance with conservation of mass. The auxiliary quantities in $\mathbf{A}_p^n$ are not relevant to our particle/grid transfers but we include them for completeness. E.g. in an MPM calculation the deformation gradient $\mathbf{F}_p^n$ is auxiliary to transfers and would be included in $\mathbf{A}_p^n$. We will generally consider the update of the auxiliary quantities to be outside the scope of the paper.

In order to update the Lagrangian state to obtain $\mathbf{x}_p^{n+1}$, $\mathbf{c}_p^{n+1}$ and $\mathbf{A}_p^{n+1}$, we first transfer mass and momentum from particle to grid (Section §2.4.1), then grid momentum is dynamically updated (Section §2.4.2) and finally, we transfer the generalized velocity information from grid to particle (Section §2.4.3). We use the notation $m_{\mathbf{i}}^n$ and $\mathbf{v}_{\mathbf{i}}^n$ to denote the mass and velocity transferred to the grid node $\mathbf{x}_{\mathbf{i}}$ from the particles before the grid momentum update. We further use the notation $\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$ to denote the grid node velocity that is updated in grid momentum update. We use this convention to distinguish it from $\mathbf{v}_{\mathbf{i}}^{n+1}$, the velocity that is transferred to the grid in the next time step. Lastly, we use $\mathbf{x}_{\mathbf{i}}^{n+1} = \mathbf{x}_{\mathbf{i}} + \Delta t \hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$ to

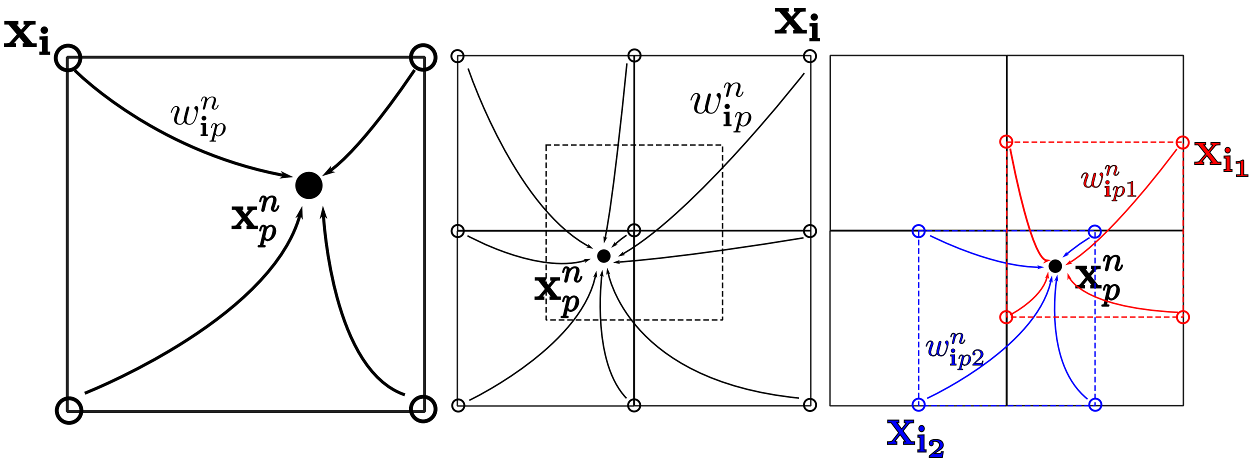

Figure 2.1: **Grid interpolation**. We visualize the weights $w_{\mathbf{i}p}^n$ for multilinear ($N_B = 1$), collocated (left), multiquadratic ($N_B = 2$), collocated (center) and weights $w_{\mathbf{i}_\alpha p}^n$ for linear ($N_B = 1$), MAC grids (right). We emphasize that the particle interpolates from $(N_B + 1)^d$ grid nodes.

denote the position of the grid nodes if they move with the grid node velocity $\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$. This process is illustrated in following commutative diagram.

$$\begin{tikzcd}
m_p,\ \mathbf{x}_p^n,\ \mathbf{c}_p^n,\ \mathbf{A}_p^n \arrow[r, "\text{Update}", "\text{Lagrangian State}"'] \arrow[d, "\text{P2G}"'] & m_p,\ \mathbf{x}_p^{n+1},\ \mathbf{c}_p^{n+1},\ \mathbf{A}_p^{n+1} \\
m_{\mathbf{i}}^n, \mathbf{v}_{\mathbf{i}}^n \arrow[r, "\text{Update}", "\text{Grid Momentum}"'] & m_{\mathbf{i}}^n, \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} \arrow[u, "\text{G2P}"']
\end{tikzcd}$$

Grid-based interpolating functions $N(\mathbf{x} - \mathbf{x_i})$ provide the mechanism for the transfer of particle and grid quantities. As in many other recent approaches [SKB08, SSC13, JSS15], the grid interpolating functions are constructed from dyadic products of one-dimensional B-splines. We use the notation $w_{\mathbf{i}p}^n = N(\mathbf{x_i} - \mathbf{x}_p^n)$ to denote the weight of interaction between node $\mathbf{x_i}$ and particle $\mathbf{x}_p^n$.

We note that a particle will interpolate from $(N_B + 1)^d$ grid nodes where N_B is the B-spline interpolating order (1 for linear, 2 for quadratic, etc) and $d = 2, 3$ is the spatial dimension. In other words, the particle with position $\mathbf{x}_p^n$ will only have non-zero weights $w_{\mathbf{i}p}^n$ for the $(N_B + 1)^d$ grid nodes most local to it. We will use the notation $\hat{\mathcal{V}}_p^{n+1} \in \mathbb{R}^{d(N_B+1)^d}$ to denote the vector of updated grid-node velocities $\hat{\mathbf{v}}_{\mathbf{i}_{kp}^n}^{n+1}$ corresponding to grid nodes $\mathbf{x}_{\mathbf{i}_{kp}^n}$

with non-zero weights $w^n_{\mathbf{i}^n_{kp}p}$

$$\hat{\mathcal{V}}^{n+1}_p = \begin{pmatrix} \hat{\mathbf{v}}^{n+1}_{\mathbf{i}^n_{1p}} \\ \hat{\mathbf{v}}^{n+1}_{\mathbf{i}^n_{2p}} \\ \vdots \\ \hat{\mathbf{v}}^{n+1}_{\mathbf{i}^n_{(N_B+1)^d p}} \end{pmatrix}.$$

We use $\mathbf{i}^n_{kp}$ for $k = 1, 2, \ldots, (N_B + 1)^d$ as an index for nodes with non-zero weights $w^n_{\mathbf{i}^n_{kp}p}$. We illustrate this in Figure 2.1. When it is clear from context, we will use either $\mathbf{i}_k, w^n_{\mathbf{i}_k p}$ or even $\mathbf{i}_k, w^n_{\mathbf{i}p}$ in lieu of the more descriptive $\mathbf{i}^n_{kp}, w^n_{\mathbf{i}^n_{kp}p}$ since the sub and super indices can become excessive in some expressions.

2.2 Velocity Modes

Our approach closely resembles that of Jiang et al. [JSS15, JST17]. Our most fundamental difference is that instead of augmenting particles with affine velocities, we augment them with more general functions. In the APIC approaches advocated by Jiang et al. [JSS15, JST17], the velocity local to the particle p at time t^n is approximated as

$$\mathbf{v}^n_p(\mathbf{x}) = \mathbf{v}^n_p + \mathbf{C}^n_p(\mathbf{x} - \mathbf{x}^n_p)$$

where $\mathbf{v}^n_p$ is the velocity of the particle and the matrix $\mathbf{C}^n_p \in \mathbb{R}^{d\times d}$ satisfies $\mathbf{C}^n_p = \mathbf{0}$ for PIC, $\mathbf{C}^n_p = -\left(\mathbf{C}^n_p\right)^T$ for RPIC (locally rigid PIC) and $\mathbf{C}^n_p$ is arbitrary for APIC.

In this paper, we improve the approach by considering the particle-wise local velocity to be of the form

$$\mathbf{v}^n_p(\mathbf{x}) = \sum_{r=1}^{N_r} \sum_{\alpha=1}^{d} s_r(\boldsymbol{\xi}^n_p(\mathbf{x}) - \mathbf{x}^{n-1}_p)\mathbf{e}_\alpha c^n_{pr\alpha} \tag{2.1}$$

where the functions $s_r\mathbf{e}_\alpha : \mathbb{R}^d \to \mathbb{R}^d$ are generalized velocity modes, $\mathbf{e}_\alpha \in \mathbb{R}^d$ is the α^{th} standard basis vector and the $c^n_{pr\alpha}$ are the coefficients of the modes which are stored in the

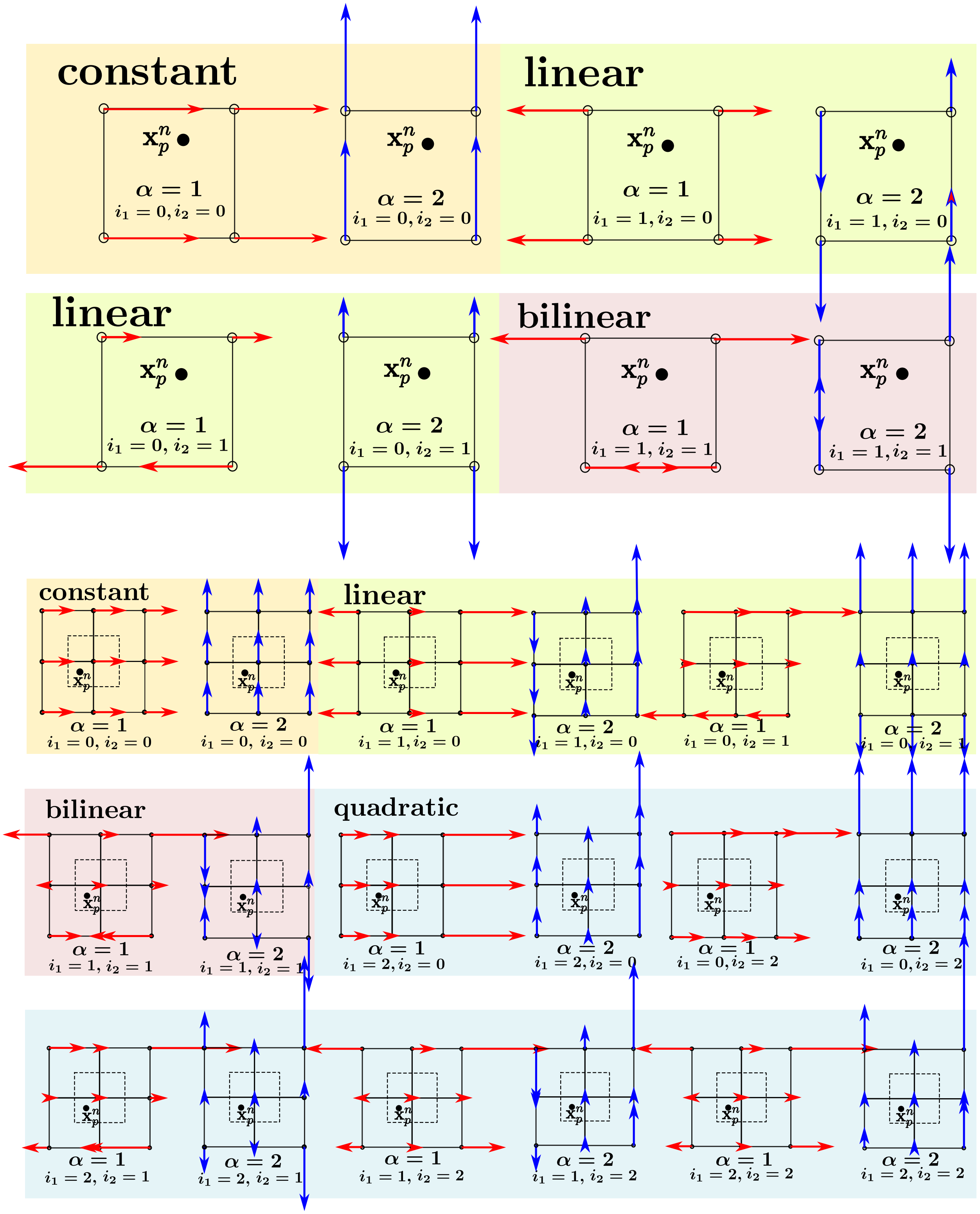

Figure 2.2: **Velocity modes.** We visualize the component-wise velocity modes from Equation (2.2) in 2D. The top shows bilinear interpolation and the bottom shows biquadratic interpolation. Constant (peach), linear (green), bilinear (pink) and biquadratic (light blue) modes are depicted for x (red) and y (blue) components.

vector $\mathbf{c}_p^n \in \mathbb{R}^{dN_r}$. We build our generalized velocity modes component-by-component in terms of the scalar functions $s_r : \mathbb{R}^d \to \mathbb{R}$. N_r indicates the total number of scalar modes that we use. We illustrate these modes in Figure 2.2. The function $\boldsymbol{\xi}_p^n$ approximates the mapping from the time t^n configuration to the time t^{n-1} configuration local to the particle and represents the advection of the material (see Section §2.3). We note that in PIC and APIC it is simply given by $\boldsymbol{\xi}_p^n(\mathbf{x}) = \mathbf{x} - \Delta t \mathbf{v}_p^{n-1}$ and $\boldsymbol{\xi}_p^n(\mathbf{x}) - \mathbf{x}_p^{n-1} = \mathbf{x} - \mathbf{x}_p^n$.

By approximating the velocity local to particle $\mathbf{x}_p^n$ in terms of more general functions, we allow for a wider range of local behavior than in the original APIC. Notably, we can write APIC in this way by choosing affine functions for s_r. Similarly, we can write PIC in this way by choosing constant functions for the s_r. In either case we note that the coefficients $\mathbf{c}_p^n \in \mathbb{R}^{dN_r}$ are equivalent to the $\mathbf{v}_p^n$ and $\mathbf{C}_p^n$ in the original APIC and PIC. Note that for APIC, $dN_r = d^2 + d$ (d translations and d^2 linear functions) and similarly for PIC, $dN_r = d$.

We primarily use polynomial modes of the form

$$s(\mathbf{z}) = \prod_{\beta=1}^{d} z_\beta^{i_\beta}. \tag{2.2}$$

Here z_β is the β^{th} component of $\mathbf{z} \in \mathbb{R}^d$, the $i_\beta \in \mathbb{Z}^+$ are non-negative integer powers. We note that this reduces to the original PIC when $i_\beta = 0$ for $1 \leq \beta \leq d$. Furthermore, when we choose all s_r with exactly one of the $i_\beta = 1$ and the rest equal to zero, we obtain the affine modes and the method reduces to APIC. In general, we will modify the polynomial modes in Equation (2.2) slightly to ensure a mass-orthogonality condition that is essential for efficiency in the grid to particle transfer (see Section §2.4.3).

The particle-wise local velocity in Equation (2.1) is used in the particle-to-grid and grid-to-particle transfers. As in [JSS15, JST17], it is used to define a particle's contribution to the grid node linear momentum in the particle-to-grid transfer (Section §2.4.1). In the grid-to-particle transfer (Section §2.4.3), the coefficients $\mathbf{c}_p^{n+1}$ are chosen so that

$$\hat{\mathbf{v}}_p^{n+1}(\mathbf{y}) = \sum_{r=1}^{N_r} \sum_{\alpha=1}^{d} s_r(\mathbf{y} - \mathbf{x}_p^n) \mathbf{e}_\alpha c_{pr\alpha}^{n+1} \approx \sum_{\mathbf{i}} \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} N(\mathbf{y} - \mathbf{x}_{\mathbf{i}}) \tag{2.3}$$

for $\mathbf{y}$ near $\mathbf{x}_p^n$. However, this approximation is done with points $\mathbf{y}$ in the time t^n rather than t^{n+1} configuration of the material. This is a local approximation of the updated Lagrangian velocity. The updated Lagrangian velocity is related to the Eulerian velocity by the mapping $\left(\boldsymbol{\xi}_p^{n+1}\right)^{-1}$, which approximates the advection of the material local to the particle to the time t^{n+1} configuration. We discuss the significance of this mapping and the notion of updated Lagrangian velocity in the next section.

2.3 Updated Lagrangian

Eulerian and Lagrangian methods can be characterized in terms of the flow map of the material $\boldsymbol{\phi}(\cdot, t) : \Omega^0 \to \Omega^t$ [GS08]. Here $\Omega^0 \subset \mathbb{R}^d$ is the initial configuration of the material. Each point $\mathbf{X} \in \Omega^0$ is the initial position of a particle of material in the continuum and $\boldsymbol{\phi}(\mathbf{X}, t)$ is its location at time t. Ω^t is the time t configuration of the material consisting of the points $\mathbf{x} = \boldsymbol{\phi}(\mathbf{X}, t)$ for some $\mathbf{X} \in \Omega^0$. It is this mapping that defines the Lagrangian velocity and acceleration of each particle via $\mathbf{V}(\mathbf{X}, t) = \frac{\partial \boldsymbol{\phi}}{\partial t}(\mathbf{X}, t)$ and $\mathbf{A}(\mathbf{X}, t) = \frac{\partial \mathbf{V}}{\partial t}(\mathbf{X}, t)$. The Eulerian counterparts can be defined in terms of the inverse of the flow map $\boldsymbol{\phi}^{-1}(\cdot, t) : \Omega^t \to \Omega^0$ via $\mathbf{v}(\mathbf{x}, t) = \mathbf{V}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t)$ and $\mathbf{a}(\mathbf{x}, t) = \mathbf{A}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t)$ for all $\mathbf{x} \in \Omega^t$. Here $\mathbf{X} = \boldsymbol{\phi}^{-1}(\boldsymbol{\phi}(\mathbf{X}, t), t)$ for all particles $\mathbf{X} \in \Omega^0$ and $\mathbf{x} = \boldsymbol{\phi}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), t)$ for all points $\mathbf{x} \in \Omega^t$. Also, the Eulerian velocity and acceleration are related through the total derivative

$$\mathbf{a}(\mathbf{x}, t) = \frac{D\mathbf{v}}{Dt}(\mathbf{x}, t) = \frac{\partial \mathbf{v}}{\partial t}(\mathbf{x}, t) + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}(\mathbf{x}, t)\mathbf{v}(\mathbf{x}, t). \tag{2.4}$$

We note that Levin et al. have developed a number of methods that make use of the inverse flow map [LLJ11, FLL13, TLK16].

In Lagrangian approaches, the initial configuration of the material Ω^0 serves as the domain of field functions like velocity and stress. This is sometimes referred to as a total Lagrangian approach. Updated Lagrangian approaches use a similar idea, but rather than using a single reference configuration Ω^0, the time t^n configuration Ω^{t^n} is used as the reference. With these approaches, it is convenient to define $\hat{\mathbf{v}}(\mathbf{y}, t) = \mathbf{V}(\boldsymbol{\phi}^{-1}(\mathbf{y}, t^n), t)$ and

$\hat{\mathbf{a}}(\mathbf{y}, t) = \mathbf{A}(\boldsymbol{\phi}^{-1}(\mathbf{y}, t^n), t)$ for all $\mathbf{y} \in \Omega^{t^n}$. These are the time t velocity and acceleration defined over Ω^{t^n}. They are analogous to the Lagrangian acceleration and velocity, except the points $\mathbf{y} \in \Omega^{t^n}$ serve as the reference for each particle in the continuum, rather than the initial points $\mathbf{X} \in \Omega^0$. This is convenient because unlike the Eulerian velocity and acceleration in Equation (2.4) that relate through the total derivative, $\hat{\mathbf{v}}$ and $\hat{\mathbf{a}}$ relate through standard temporal differentiation

$$\hat{\mathbf{a}}(\mathbf{y}, t) = \frac{\partial \hat{\mathbf{v}}}{\partial t}(\mathbf{y}, t). \tag{2.5}$$

Thus, the updated Lagrangian velocity and acceleration have the same essential relation as their total Lagrangian counterparts.

PIC can be viewed as an updated Lagrangian approach. At time t^n the particles have positions $\mathbf{x}_p^n = \boldsymbol{\phi}(\mathbf{X}_p, t^n)$ and represent samples of Ω^{t^n}. When we transfer state to the grid (see Section §2.4.2), we obtain approximations to the velocity $\mathbf{v}_{\mathbf{i}}^n$ and mass $m_{\mathbf{i}}^n$ at grid nodes $\mathbf{x}_{\mathbf{i}} \in \Omega^{t^n}$. This provides an alternative approximation to the time t^n configuration that has the advantage of being defined over particles with structured (grid-aligned) locations, as opposed to the unstructured $\mathbf{x}_p^n$. The structured nature of their locations has many advantages, e.g. it is easy to interpolate data via regular grid interpolating functions. Indeed, the Eulerian velocity at time t^n can be approximated via interpolation from $\mathbf{v}(\mathbf{x}, t^n) \approx \sum_{\mathbf{i}} \mathbf{v}_{\mathbf{i}}^n N(\mathbf{x} - \mathbf{x}_{\mathbf{i}})$. In the grid momentum update step, we assume that Ω^{t^n} is the updated reference configuration and approximate the updated Lagrangian acceleration in an essentially Lagrangian manner via

$$\hat{\mathbf{a}}(\mathbf{y}, t^{n+1}) \approx \sum_{\mathbf{i}} \frac{\hat{\mathbf{v}}_{\mathbf{i}}^{n+1} - \mathbf{v}_{\mathbf{i}}^n}{\Delta t} N(\mathbf{y} - \mathbf{x}_{\mathbf{i}}),\ \mathbf{y} \in \Omega^{t^n}.$$

Here, the new grid node velocities $\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$ approximate samples of $\hat{\mathbf{v}}(\mathbf{y}, t^{n+1})$ at grid nodes $\mathbf{y} = \mathbf{x}_{\mathbf{i}} \in \Omega^{t^n}$. Their interpolant approximates the updated Lagrangian velocity $\hat{\mathbf{v}}(\mathbf{y}, t^{n+1}) = \sum_{\mathbf{i}} \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} N(\mathbf{y} - \mathbf{x}_{\mathbf{i}})$, i.e. the time t^{n+1} velocity but defined over $\mathbf{y} \in \Omega^{t^n}$.

By definition, the time t^{n+1} Eulerian velocity is related to the updated Lagrangian velocity through

$$\mathbf{v}(\mathbf{x}, t^{n+1}) = \hat{\mathbf{v}}(\boldsymbol{\xi}^{n+1}(\mathbf{x}), t^{n+1}) \tag{2.6}$$

for $\mathbf{x} \in \Omega^{t^{n+1}}$. Here we use $\boldsymbol{\xi}^{n+1}(\mathbf{x}) = \boldsymbol{\phi}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t^{n+1}), t^n)$ to denote the mapping of material from the time t^{n+1} configuration to the time t^n configuration. Intuitively, $\boldsymbol{\phi}^{-1}$ maps the point $\mathbf{x} \in \Omega^{t^{n+1}}$ to its reference location $\mathbf{X} = \boldsymbol{\phi}^{-1}(\mathbf{x}, t^{n+1}) \in \Omega^0$ and $\boldsymbol{\phi}$ maps the reference location to the its position $\boldsymbol{\xi}^{n+1}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{X}, t^n) \in \Omega^{t^n}$. Thus $\mathbf{X} = \boldsymbol{\phi}^{-1}(\mathbf{x}, t^{n+1}) \in \Omega^0$ is the location in the reference configuration Ω^0 of the particle that occupies $\mathbf{x} \in \Omega^{t^{n+1}}$ and $\boldsymbol{\xi}^{n+1}(\mathbf{x}) = \boldsymbol{\phi}(\boldsymbol{\phi}^{-1}(\mathbf{x}, t^{n+1}), t^n)$ is the location in the time t^n configuration Ω^{t^n} of the particle that occupies $\mathbf{x} \in \Omega^{t^{n+1}}$. In other words, the mapping reverses the motion of material over the time step. Since the updated Lagrangian velocity is the time t^{n+1} velocity, defined over the time t^n configuration, the composition with this mapping in Equation (2.6) can be viewed as the key to defining the Eulerian velocity in a PIC calculation.

2.3.1 Particle-Wise Velocity Modes

The $\hat{\mathbf{v}}_p^{n+1}(\mathbf{y})$ from Equation (2.3) locally approximate the updated Lagrangian velocity $\hat{\mathbf{v}}(\mathbf{y}, t^{n+1})$ for $\mathbf{y} \in \Omega^{t^n}$ near $\mathbf{x}_p^n$. We use them to obtain the similar, but more useful approximation to the Eulerian velocity $\mathbf{v}_p^{n+1}(\mathbf{x})$ to $\mathbf{v}(\mathbf{x}, t^{n+1})$. The $\mathbf{v}_p^{n+1}(\mathbf{x})$ are required for the transfers from particle to grid at the beginning of time step t^{n+1} (see Section §2.4.1). We obtain them by composition with the local approximation $\boldsymbol{\xi}_p^{n+1}(\mathbf{x})$ to $\boldsymbol{\xi}^{n+1}(\mathbf{x})$ for $\mathbf{x} \in \Omega^{t^{n+1}}$ near $\mathbf{x}_p^{n+1}$

$$\mathbf{v}_p^{n+1}(\mathbf{x}) = \hat{\mathbf{v}}_p^{n+1}(\boldsymbol{\xi}_p^{n+1}(\mathbf{x})). \tag{2.7}$$

In our approach, as well as in PIC and APIC, particles move with the interpolated updated Lagrangian velocity

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \sum_{\mathbf{i}} \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} N(\mathbf{x}_p^n - \mathbf{x}_{\mathbf{i}}).$$

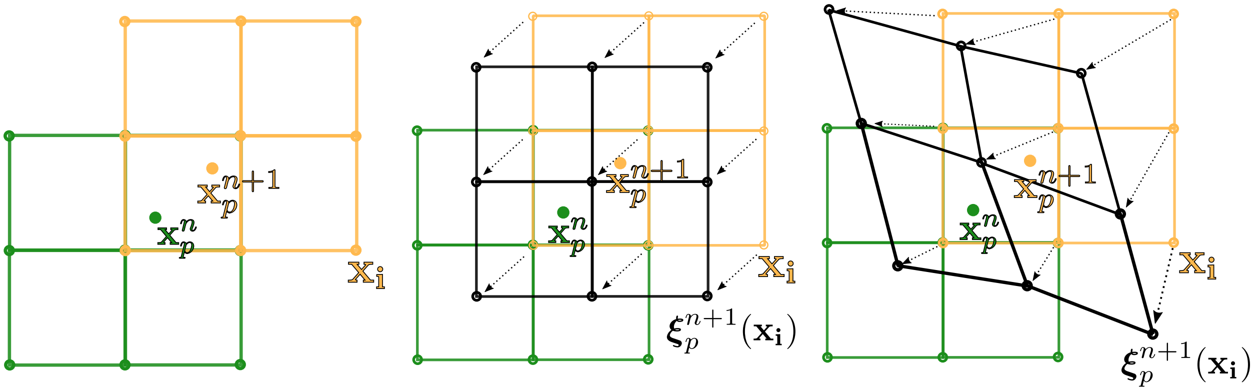

Figure 2.3: **Updated Lagrangian.** Here we visualize the options for the mapping $\boldsymbol{\xi}_p^{n+1}$. As a particle moves, from $\mathbf{x}_p^n$ to $\mathbf{x}_p^{n+1}$, it selects different grid nodes $\mathbf{x_i}$ to biquadratically interpolate from (green at t^n and yellow at t^{n+1}). The middle shows the $\boldsymbol{\xi}_p^{n+1}(\mathbf{x_i})$ approximation from Equation (2.8) and the right from Equation (2.10).

This motion of the particles defines the material mapping from configuration Ω^{t^n} to configuration $\Omega^{t^{n+1}}$. Since $\boldsymbol{\xi}^{n+1}(\mathbf{x})$ is the inverse of this mapping, we know its value at each particle $\boldsymbol{\xi}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{x}_p^n$. We can use this to approximate the mapping local to each particle.

2.3.1.1 Piecewise constant material motion

If we assume that $\boldsymbol{\xi}^{n+1}$ is approximately a simple translation near $\mathbf{x}_p^{n+1}$ and that $\boldsymbol{\xi}^{n+1}(\mathbf{x}_p^{n+1}) = \mathbf{x}_p^n$, then we obtain the local approximation

$$\boldsymbol{\xi}_p^{n+1}(\mathbf{x}) = \mathbf{x}_p^n + \left(\mathbf{x} - \mathbf{x}_p^{n+1}\right). \tag{2.8}$$

The PIC and APIC transfers can be derived from the local velocities in Equation (2.1) combined with the advection approximation in Equation (2.8). With PIC, the local updated Lagrangian velocity is constant $\hat{\mathbf{v}}_p^{n+1}(\mathbf{y}) = \mathbf{v}_p^{n+1}$ and thus for any local approximation $\boldsymbol{\xi}_p^{n+1}(\mathbf{x})$,

$$\mathbf{v}_{\text{PIC},p}^{n+1}(\mathbf{x}) = \mathbf{v}_p^{n+1}.$$

With APIC, the local updated Lagrangian velocity is affine $\hat{\mathbf{v}}_p^{n+1}(\mathbf{y}) = \mathbf{v}_p^{n+1} + \mathbf{C}_p^{n+1}(\mathbf{y} - \mathbf{x}_p^n)$. If we combine this affine velocity via the composition in Equation (2.8) with the constant local approximation in Equation (2.7), we obtain

$$\mathbf{v}_{\mathrm{APIC},p}^{n+1}(\mathbf{x}) = \hat{\mathbf{v}}_p^{n+1}(\mathbf{x}_p^n + \left(\mathbf{x} - \mathbf{x}_p^{n+1}\right)) = \mathbf{v}_p^{n+1} + \mathbf{C}_p^{n+1}(\mathbf{x} - \mathbf{x}_p^{n+1}).$$

2.3.1.2 Piecewise affine material motion

It is evident that more accurate local approximations to $\boldsymbol{\xi}^{n+1}(\mathbf{x})$ are readily available. If we assume that the updated Lagrangian velocity is well approximated by $\hat{\mathbf{v}}_p^{n+1}(\mathbf{y})$ for particles $\mathbf{y} \in \Omega^{t^n}$ near $\mathbf{x}_p^n$, then particle trajectories will evolve locally as approximately $\mathbf{y}^{n+1} = \mathbf{y} + \Delta t \hat{\mathbf{v}}_p^{n+1}(\mathbf{y})$. This approximates the motion of the material from Ω^{t^n} to $\Omega^{t^{n+1}}$ for particles near $\mathbf{x}_p^n$. The inverse of the mapping local to the particle is then approximately given by $\boldsymbol{\xi}_p^{n+1}(\mathbf{x}) = \hat{\mathbf{x}}$ where $\hat{\mathbf{x}}$ is given by the solution to the implicit equation

$$\mathbf{x} = \hat{\mathbf{x}} + \Delta t \hat{\mathbf{v}}_p^{n+1}(\hat{\mathbf{x}}). \tag{2.9}$$

Intuitively, for particle $\mathbf{x} \in \Omega^{t^{n+1}}$, $\hat{\mathbf{x}} \in \Omega^{t^n}$ is its location at time t^n. For general functions $\hat{\mathbf{v}}_p^{n+1}$, Equation (2.9) can be solved using Newton's method; however if we approximate the updated Lagrangian velocity by its affine components $\hat{\mathbf{v}}_p^{n+1}(\hat{\mathbf{x}}) \approx \sum_{r=1}^{N_r} \sum_{\alpha=1}^{d} s_r(\hat{\mathbf{x}} - \mathbf{x}_p^n)\mathbf{e}_\alpha c_{pr\alpha}^{n+1}$ with $c_{pr\alpha}^{n+1}$ only non-zero for affine modes, then the system for $\hat{\mathbf{x}}$ is linear and we obtain

$$\boldsymbol{\xi}_p^{n+1}(\mathbf{x}) = \mathbf{x}_p^n + \left(\mathbf{I} + \Delta t \mathbf{C}_p^{n+1}\right)^{-1} \left(\mathbf{x} - \mathbf{x}_p^{n+1}\right) \tag{2.10}$$

where $\mathbf{C}_p^{n+1}$ is the linear part of the polynomial modes. When $\mathbf{C}_p^{n+1} = \mathbf{0}$, we obtain the constant approximation in Equation (2.8) and thus this can be seen as a higher-order approximation. We visualize the approximations to $\boldsymbol{\xi}_p^{n+1}$ in Equations (2.8) and (2.10) in Figure 2.3.

2.4 Method

Here we detail all of the steps necessary for advancing the Lagrangian state from time t^n to t^{n+1} in a PIC calculation. We cover the necessary details for a MPM approach with elastoplastic materials as well as for incompressible Euler fluids with pressure projection on a MAC grid. This process consists of (1) the transfer from particle to grid of mass and linear momentum (Section §2.4.1), (2) the grid based momentum update (Section §2.4.2) and (3) the transfer from grid to particle of generalized velocity coefficients (Section §2.4.3). The particle-wise local approximations discussed in Sections §2.2 and §2.3 are the keys to the particle/grid transfers. We provide the details of the grid momentum update for completeness but we note that it is not novel as our approach is not relevant to this step.

2.4.1 Transfer from Particle to Grid

The velocity local to the particle $\mathbf{v}_p^n : \mathbb{R}^d \to \mathbb{R}^d$ from Equation (2.1) is used to design the momentum transfer to the grid. We use the notation $(m\mathbf{v})_{\mathbf{i}p}^n = m_p w_{\mathbf{i}p}^n \mathbf{v}_p^n(\mathbf{x_i})$ to denote the particle's contribution to the momentum local to the node $\mathbf{x_i}$ and $(m\mathbf{v})_{\mathbf{i}}^n = \sum_p (m\mathbf{v})_{\mathbf{i}p}^n$ is the total momentum of grid node from the contribution of all particles. Similarly, the contribution of the particle's mass to the grid node $\mathbf{x_i}$ is $m_{\mathbf{i}p}^n = w_{\mathbf{i}p}^n m_p$ and the total grid node mass is the sum of the contributions from all particles $m_{\mathbf{i}}^n = \sum_p m_{\mathbf{i}p}^n$. Using this we can define the grid node velocity $\mathbf{v}_{\mathbf{i}}^n$ by dividing momentum by mass. In summary, this transfer consists of

$$\begin{aligned}(m\mathbf{v})_{\mathbf{i}p}^n &= m_{\mathbf{i}p}^n \sum_{r=1}^{N_r} \sum_{\alpha=1}^{d} s_r(\boldsymbol{\xi}_p^n(\mathbf{x_i}) - \mathbf{x}_p^{n-1}) \mathbf{e}_\alpha c_{pr\alpha}^n \\ (m\mathbf{v})_{\mathbf{i}}^n &= \sum_p (m\mathbf{v})_{\mathbf{i}p}^n, \ \ \mathbf{v}_{\mathbf{i}}^n = \frac{(m\mathbf{v})_{\mathbf{i}}^n}{m_{\mathbf{i}}^n}.\end{aligned} \tag{2.11}$$

Either local approximations $\boldsymbol{\xi}_p^n(\mathbf{x_i})$ from Equations (2.8) or (2.10) can be used. We note that this is essentially the same transfer as in the original APIC approaches [JSS15, JST17], with the only modification being the more general notions of the local velocity and the improved

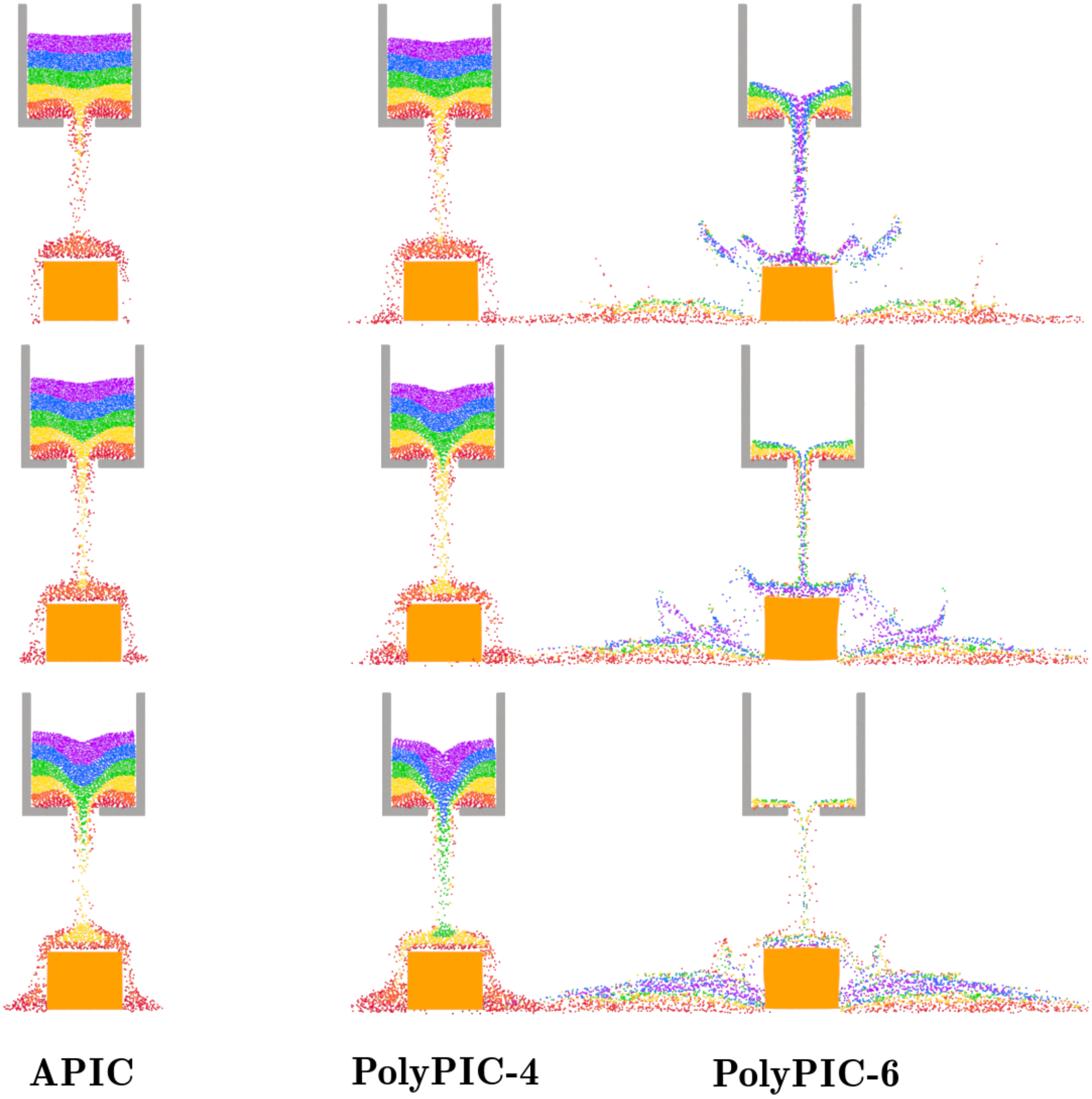

Figure 2.4: **MPM elastoplasticity**. Rainbow colored sand is poured onto an elastic Jell-O square. We compare APIC (left) vs. PolyPIC with (from left to right) $N_r = 4$ and $N_r = 6$. Notice that increasing degrees of PolyPIC allow for more energetic sand flowing and Jell-O bouncing.

approximation to $\boldsymbol{\xi}_p^n(\mathbf{x}_\mathbf{i})$.

2.4.2 Update Grid Momentum

The grid momentum update is outside the scope of this paper. However, we include a generic description for representative cases that we used to generate our examples: incompressible Euler fluids and elastoplastic solids with MPM. In the case of the incompressible Euler, we used a MAC grid discretization of the pressure projection to update the fluid velocity. In the case of elastoplastic solids and MPM the update is from the elastic force (see [FGG17a] for more details).

$$\hat{\mathbf{v}}_\mathbf{i}^{n+1} = \mathbf{v}_\mathbf{i}^n + \frac{\Delta t}{\rho}\nabla p, \qquad \text{(Euler/MAC)}$$
$$\hat{\mathbf{v}}_\mathbf{i}^{n+1} = \mathbf{v}_\mathbf{i}^n + \frac{\Delta t}{m_\mathbf{i}^n}(\mathbf{f} + \mathbf{g}), \qquad \text{(elastoplastic/MPM)}$$

where $\mathbf{f}$ is the elastic force and $\mathbf{g}$ is the gravitational acceleration.

2.4.3 Transfer from Grid to Particle

The transfer from grid to particle is achieved by choosing the generalized velocity coefficients $\mathbf{c}_p^{n+1} \in \mathbb{R}^{dN_r}$ so that the approximation in Equation (2.3) is optimal in the appropriate sense. Here we show that we can solve a linear system for the coefficients $\mathbf{c}_p^{n+1} \in \mathbb{R}^{dN_r}$, and that by design our approach (1) is equivalent to PIC and APIC if only constant or affine modes are used, (2) conserves linear and angular momentum (see [FGG17a]) and (3) has a *diagonal* system matrix in the equation for the $\mathbf{c}_p^{n+1} \in \mathbb{R}^{dN_r}$.

We choose the coefficients $\mathbf{c}_p^{n+1}$ to minimize the mass-weighted distance $d_p^n(\mathbf{c}_p^{n+1})$ between local velocities at the grid nodes and the updated grid-node velocities

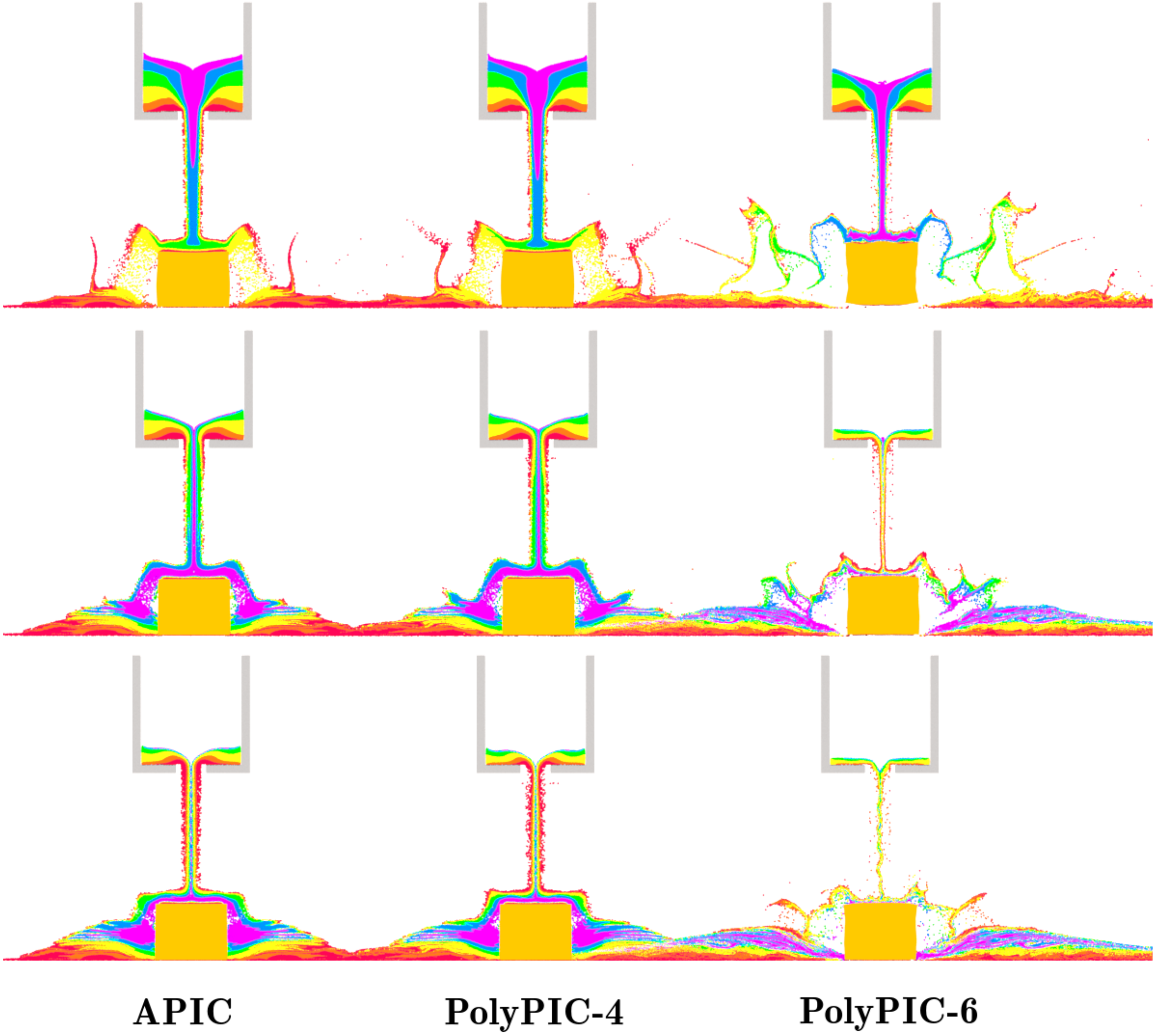

Figure 2.5: **MPM elastoplasticity refinement**. We verify that the behavior exhibited by PolyPIC with $N_r = 6$ at lower resolution in Figure 2.4 is exhibited by PolyPIC with $N_r = 4$ and APIC under refinement.

s_t s_r	1	x	y	xy	x^2	y^2	x^2y	xy^2	x^2y^2
1	X				X	X			X
x		X			X			X	X
y			X			X	X		X
xy				X			X	X	X
x^2	X	X			X	X		X	X
y^2	X		X		X	X	X		X
x^2y			X	X		X	X	X	X
xy^2		X		X	X		X	X	X
x^2y^2	X	X	X	X	X	X	X	X	X

Table 2.1: **Sparsity pattern: unmodified.** We illustrate the sparsity pattern of the matrix $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n$ for dimension $d = 2$ with scalar modes $s = x^{i_1} y^{i_2}$. X indicates a non-zero entry in the matrix. Note that the multilinear modes (indicated in red) are mass-orthogonal to one another, but that the multiquadratic modes couple extensively.

$$\begin{aligned} d_p^n(\mathbf{c}_p^{n+1}) &= \sum_{\mathbf{i}} m_{\mathbf{i}p}^n \left| \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} - \hat{\mathbf{v}}_p^{n+1}(\mathbf{x}_{\mathbf{i}}) \right|^2 \\ &= \sum_{\mathbf{i}} m_{\mathbf{i}p}^n \left| \hat{\mathbf{v}}_{\mathbf{i}}^{n+1} - \sum_{r=1}^{N_r} \sum_{\alpha=1}^{d} s_r(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n) \mathbf{e}_\alpha c_{pr\alpha}^{n+1} \right|^2 \end{aligned}$$

where $m_{\mathbf{i}p}^n = m_p w_{\mathbf{i}p}^n$ is the mass that the particle $\mathbf{x}_p^n$ transfers to the grid node $\mathbf{i}$. The minimizer of this mass weighted distance can be expressed more concisely in terms of the grid node locations that received non-zero mass from the particle $\mathbf{x}_p^n$. Recall that the particle will have non-zero weights $w_{\mathbf{i}p}^n$ for precisely the $(N_B+1)^d$ grid nodes in closest proximity to the particle and that $\hat{\mathcal{V}}_p^{n+1} \in \mathbb{R}^{d(N_B+1)^d}$ is the vector of velocities of the grid nodes with non-zero weights. Similarly, we use the notation $\mathbf{Q}_p^n = \left[\mathcal{Q}_{p11}^n, \mathcal{Q}_{p12}^n, \dots, \mathcal{Q}_{pN_rd}^n\right] \in \mathbb{R}^{d(N_B+1)^d \times dN_r}$ where the columns $\mathcal{Q}_{pr\alpha}^n$ of $\mathbf{Q}_p^n$ are analogous to $\hat{\mathcal{V}}_p^{n+1}$ and have entries equal to the particle-wise

s_t s_r	1	x	y	xy	$g_1(x)$	$g_2(y)$	$g_1(x)y$	$xg_2(y)$	$g_1(x)g_2(y)$
1	a								
x		b							
y			b						
xy				c					
$g_1(x)$					$d(x)$				
$g_2(y)$						$d(y)$			
$g_1(x)y$							$e(x)$		
$xg_2(y)$								$e(y)$	
$g_1(x)g_2(y)$									$f(x,y)$

Table 2.2: **Sparsity pattern: modified.** We illustrate the sparsity pattern of the matrix $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n$ for dimension $d = 2$ with the modified quadratic modes given by Equation (2.14). $a = 1$, $b = \frac{\Delta x^2}{4}$, $c = \frac{\Delta x^2}{16}$, $d(z) = \frac{(\Delta x^2 - 4z^2)^2(3\Delta x^2 - 4z^2)}{16\Delta x^2}$, $e(z) = \frac{(\Delta x^2 - 4z^2)^2(3\Delta x^2 - 4z^2)}{64}$, $f(x,y) = \frac{(\Delta x^2 - 4x^2)^2(3\Delta x^2 - 4x^2)(\Delta x^2 - 4y^2)^2(3\Delta x^2 - 4y^2)}{256\Delta x^4}$.

local modes $s_r(\mathbf{x_i} - \mathbf{x}_p^n)\mathbf{e}_\alpha$ at the grid nodes with non-zero weights

$$\mathcal{Q}_{pr\alpha}^n = \begin{pmatrix} s_r(\mathbf{x}_{\mathbf{i}_1} - \mathbf{x}_p^n)\mathbf{e}_\alpha \\ s_r(\mathbf{x}_{\mathbf{i}_2} - \mathbf{x}_p^n)\mathbf{e}_\alpha \\ \vdots \\ s_r(\mathbf{x}_{\mathbf{i}_{(N_B+1)^d}} - \mathbf{x}_p^n)\mathbf{e}_\alpha \end{pmatrix}.$$

The optimal coefficients $\mathbf{c}_p^{n+1}$ can be expressed in terms of these vectors as

$$\mathbf{c}_p^{n+1} = \underset{\mathbf{c} \in \mathbb{R}^{dN_r}}{\text{argmin}} \; d_p^n(\mathbf{c}) = \left((\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n\right)^{-1} (\mathbf{Q}_p^n)^T \mathbf{M}_p^n \hat{\mathcal{V}}_p^{n+1} \tag{2.12}$$

where the matrix $\mathbf{M}_p^n \in \mathbb{R}^{d(N_B+1)^d \times d(N_B+1)^d}$ is diagonal and consists of $(N_B + 1)$ diagonal blocks $m_{\mathbf{i}p}^n \mathbf{I}_d$. Here $\mathbf{I}_d \in \mathbb{R}^{d \times d}$ is the $d-$dimensional identity.

2.4.3.1 Dimension-by-dimension decoupling

Our approach is only efficient if the linear system for $\mathbf{c}_p^{n+1}$ in Equation (2.12) can be solved quickly. Fortunately, the matrix $(\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n \in \mathbb{R}^{dN_r \times dN_r}$ has remarkable properties for polynomial velocity modes of the type in Equation (2.2). First, because the matrix $\mathbf{M}_p^n$ is

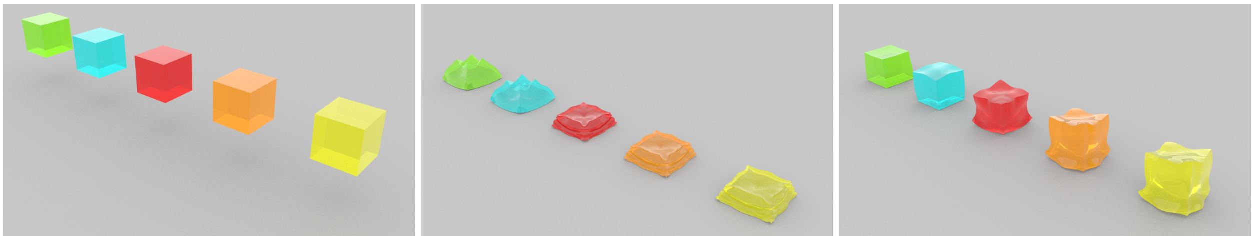

Figure 2.6: **MPM hyperelasticity**. We compare from left to right APIC (green) and PolyPIC with $N_r = 8$ (blue), $N_r = 11$ (red), $N_r = 14$ (orange), $N_r = 18$ (yellow). PolyPIC better conserves total energy which results in less numerical damping of the deformable motion.

diagonal,

$$\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n = \begin{pmatrix} m_{\mathbf{i}_1 p}^n s_t(\mathbf{x}_{\mathbf{i}_1} - \mathbf{x}_p^n)\mathbf{e}_\beta \\ m_{\mathbf{i}_2 p}^n s_t(\mathbf{x}_{\mathbf{i}_2} - \mathbf{x}_p^n)\mathbf{e}_\beta \\ \vdots \\ m_{\mathbf{i}_{(N_B+1)^d} p}^n s_t(\mathbf{x}_{\mathbf{i}_{(N_B+1)^d}} - \mathbf{x}_p^n)\mathbf{e}_\beta \end{pmatrix}.$$

Therefore, the entries in $\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n$ are proportionate to the entries in $\mathcal{Q}_{pr\alpha}^n$. Thus, since they have the same dimension-by-dimension sparsity as a consequence of the $\mathbf{e}_\beta$ and $\mathbf{e}_\alpha$ terms, the individual dimensions decouple when we take the dot products $\mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n\right)$. The exact expression for the entries in the matrix $(\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n$ are then

$$\mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n\right) = \sum_{\mathbf{i}} m_{\mathbf{i}}^n s_r(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n) s_t(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_p^n)\mathbf{e}_\alpha \cdot \mathbf{e}_\beta.$$

Here we use r, α to index the rows and t, β to index the columns of the matrix $(\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n$. Furthermore, r, t index the mode type while α, β index the dimension. Therefore the matrix entries $\mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n\right) = 0$ when $\alpha \neq \beta$ since $\mathbf{e}_\alpha \cdot \mathbf{e}_\beta = 0$ when $\alpha \neq \beta$. Thus the coefficients $c_{pr\alpha}^n$ are decoupled in α and the matrix $(\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n$ is block diagonal with d identical diagonal blocks associated with the dimension-by-dimension velocity modes.

The d non-zero diagonal blocks of $(\mathbf{Q}_p^n)^T \mathbf{M}_p^n \mathbf{Q}_p^n \in \mathbb{R}^{dN_r \times dN_r}$ are each equal to $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n \in \mathbb{R}^{N_r \times N_r}$. Here $\mathbf{m}_p^n \in \mathbb{R}^{(N_B+1)^d \times (N_B+1)^d}$ is diagonal with entries equal to $m_{\mathbf{i}p}^n$. Furthermore, the matrix $\mathbf{S}_p^n$ consists of columns analogous to $\hat{\mathcal{V}}_p^{n+1}$ and $\mathcal{Q}_{pr\alpha}^n$, but with entries equal to the *scalar* particle-wise local modes $s_r(\mathbf{x_i} - \mathbf{x}_p^n)$ at the grid nodes with non-zero weights. $\mathbf{S}_p^n = \left[\mathcal{S}_{p1}^n, \ldots, \mathcal{S}_{pN_r}^n\right] \in \mathbb{R}^{(N_B+1)^d \times N_r}$ with columns

$$\mathcal{S}_{pr}^n = \begin{pmatrix} s_r(\mathbf{x}_{\mathbf{i}_1} - \mathbf{x}_p^n) \\ s_r(\mathbf{x}_{\mathbf{i}_2} - \mathbf{x}_p^n) \\ \vdots \\ s_r(\mathbf{x}_{\mathbf{i}_{(N_B+1)^d}} - \mathbf{x}_p^n) \end{pmatrix} \in \mathbb{R}^{(N_B+1)^d}.$$

With this convention, $\mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \mathcal{Q}_{pt\beta}^n\right) = \mathcal{S}_{pr}^n \cdot \left(\mathbf{m}_p^n \mathcal{S}_{pt}^n\right) \mathbf{e}_\alpha \cdot \mathbf{e}_\beta$ and the dimension-by-dimension decoupled equations for the optimal coefficients $\mathbf{c}_p^{n+1}$ are

$$\begin{aligned} \sum_{t=1}^{N_r} \mathcal{S}_{pr}^n \cdot \left(\mathbf{m}_p^n \mathcal{S}_{pt}^n\right) c_{pt\alpha}^{n+1} &= \mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \hat{\mathcal{V}}_p^{n+1}\right) \\ &= \sum_{\mathbf{i}} m_{\mathbf{i}p}^n s_r(\mathbf{x_i} - \mathbf{x}_p^n) \hat{v}_{\mathbf{i}\alpha}^{n+1} \end{aligned} \tag{2.13}$$

for $1 \leq r \leq N_r$, where $\hat{v}_{\mathbf{i}\alpha}^{n+1}$ is the α^{th} component of $\hat{\mathbf{v}}_{\mathbf{i}}^{n+1}$.

2.4.3.2 Mass-orthogonal polynomial modes

The individual blocks $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n \in \mathbb{R}^{N_r \times N_r}$ have further favorable sparsity structure. If we assume that we number the modes with increasing degree (e.g. in 2D, constant modes first: $s_1 = 1$, followed by linear $s_2 = x, s_3 = y$, then multilinear: $s_4 = xy$, etc) and if we use modes s_r with $r \leq N_r \leq 2^d$, the matrix $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n$ is diagonal. This can be verified directly using Mathematica [Res16] and we provide Mathematica code in Appendix §A.6. Notably, this means that constant modes ($r \leq 1$), linear modes ($1 < r \leq d$) and multilinear modes ($d < r \leq 2^d$) are mass-orthogonal and therefore the coefficients in Equation (2.13) can be obtained through the solution of a diagonal system.

In general for $2^d < r \leq N_r \leq (N_B + 1)^d$, the matrix $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n$ is not diagonal. We illustrate this in Table 2.1 with $d = 2$ for brevity. However, we can obtain a diagonal system with a modified Gram-Schmidt approach that takes into account the inner product defined by $\mathbf{m}_p^n$. This amounts to simple modifications of the quadratic scalar modes $2^d < r \leq N_r \leq (N_B+1)^d$ in Equation (2.2). Remarkably, the Gram-Schmidt mass-orthogonalization does not modify any of the constant, linear or multilinear modes. Only the quadratic modes are modified and the change is very simple: each quadratic term z_β^2 in Equation (2.2) is replaced with $g_\beta(z_\beta)$ given by

$$g_\beta(w) = w^2 - \frac{x_{p\beta}^n \left(\Delta x^2 - 4(x_{p\beta}^n)^2\right)}{\Delta x^2} w - \frac{\Delta x^2}{4}. \tag{2.14}$$

E.g. the mode $s_5 = g_1(x)$ replaces x^2, $s_6 = g_2(y)$ replaces y^2, $s_7 = g_1(x)y$ replaces x^2y, etc. This trivial modification yields a diagonal $(\mathbf{S}_p^n)^T \mathbf{m}_p^n \mathbf{S}_p^n$ whose entries we enumerate in Table 2.2. We give expressions for the individual entries in the solution $\mathbf{c}_p^{n+1}$ to Equation (2.13) with diagonal basis in Appendix §A.2.

We note that $(N_B+1)^d$ is a natural upper bound on the number of reduces modes N_r since the minimization in Equation (2.12) is over determined for $N_r > (N_B+1)^d$.

2.5 MAC Grid

For clarity of exposition, we only consider the case of collocated grids in Sections §2.4.1 and §2.4.3. For incompressible Euler we transfer to and from staggered velocity MAC grids [HW65]. Using $\mathbf{i}_\alpha$, $1 \leq \alpha \leq d$ to denote the face index for each of the staggered grids, MAC transfers are done component-wise (see Figure 2.1). Particle $\mathbf{x}_p^n$ transfers mass $m_{\mathbf{i}_\alpha p}^n$ to each α face grid from $m_{\mathbf{i}_\alpha p}^n = m_p^n w_{\mathbf{i}_\alpha p}^n$. The total mass on each grid face $m_{\mathbf{i}_\alpha}^n$ is equal to the sum of the contribution from each particle $m_{\mathbf{i}_\alpha}^n = \sum_p m_{\mathbf{i}_\alpha p}^n$. The weight of interaction $w_{\mathbf{i}_\alpha p}^n = N(\mathbf{x}_{\mathbf{i}_\alpha} - \mathbf{x}_p^n)$ is between the particle $\mathbf{x}_p^n$ and the MAC face $\mathbf{x}_{\mathbf{i}_\alpha}$. The component-wise particle-to-grid momentum transfer is

$$\begin{aligned} (mv)_{\mathbf{i}_\alpha p}^n &= m_{\mathbf{i}_\alpha p}^n \sum_r s_r(\boldsymbol{\xi}_p^n(\mathbf{x}_{\mathbf{i}_\alpha}) - \mathbf{x}_p^{n-1}) c_{pr\alpha}^n \\ (mv)_{\mathbf{i}_\alpha}^n &= \sum_p (mv)_{\mathbf{i}_\alpha p}^n, \; v_{\mathbf{i}_\alpha}^n = \frac{(mv)_{\mathbf{i}_\alpha}^n}{m_{\mathbf{i}_\alpha}^n}. \end{aligned} \tag{2.15}$$

These transfers are very similar to those in Equation (2.11); however each face grid gets its own mass and respective component of the momentum. This is slightly more costly since mass must be transferred d times with the MAC grid instead of just one with the collocated grid.

The transfers from grid to particle are also trivially done component-wise since as discussed in Section §2.4.3.1, the system in Equation (2.13) decouples component-wise. However, unlike in the collocated case, the mass matrix and scalar mode vectors will be different on each of the velocity face grids. We use the notation $\mathbf{m}^n_{p\alpha}$ and $\mathcal{S}^n_{pt\alpha}$ to denote this, where the appearance of α emphasizes that they vary with each face grid. With this convention we can write the system for the reduced mode components $\mathbf{c}_p^{n+1} \in \mathbb{R}^{dN_r}$ as

$$\sum_{t=1}^{N_r} \mathcal{S}^n_{pr\alpha} \cdot \left(\mathbf{m}^n_{p\alpha}\mathcal{S}^n_{pt\alpha}\right) c^{n+1}_{pt\alpha} = \sum_{k=\mathbf{i}_\alpha} m^n_{\mathbf{i}_\alpha p} s_r(\mathbf{x}_{\mathbf{i}_\alpha} - \mathbf{x}^n_p)\hat{v}^{n+1}_{\mathbf{i}_\alpha} \tag{2.16}$$

for $1 \leq r \leq N_r$. These systems are very similar to those in Equation (2.13). However, in the case of the MAC grid the matrices that appear in the left-hand side (whose entries $\mathcal{S}^n_{pr\alpha} \cdot \left(\mathbf{m}^n_{p\alpha}\mathcal{S}^n_{pt\alpha}\right)$ are indexed by $1 \leq r, t \leq N_r$) of Equation (2.16) are different on each of the α grids. In Equation 2.13 there is only one matrix on the left-hand side, independent of α.

2.6 Results

We demonstrate our method on a number of examples with incompressible flow and MPM elastoplasticity. We compare PolyPIC with APIC and FLIP in a number of representative scenarios. All incompressible flow simulations were done using Manta Flow [TP16]. In a few of our incompressible examples, we use passive advected particles as a post-process to aid in visualization. We note that these are simply advected in the flow for post-process visualization and do not use PolyPIC transfers. Also, all grid interpolation is multilinear for the incompressible flow examples. All grid interpolation is multiquadratic for the MPM elastoplasticity examples.

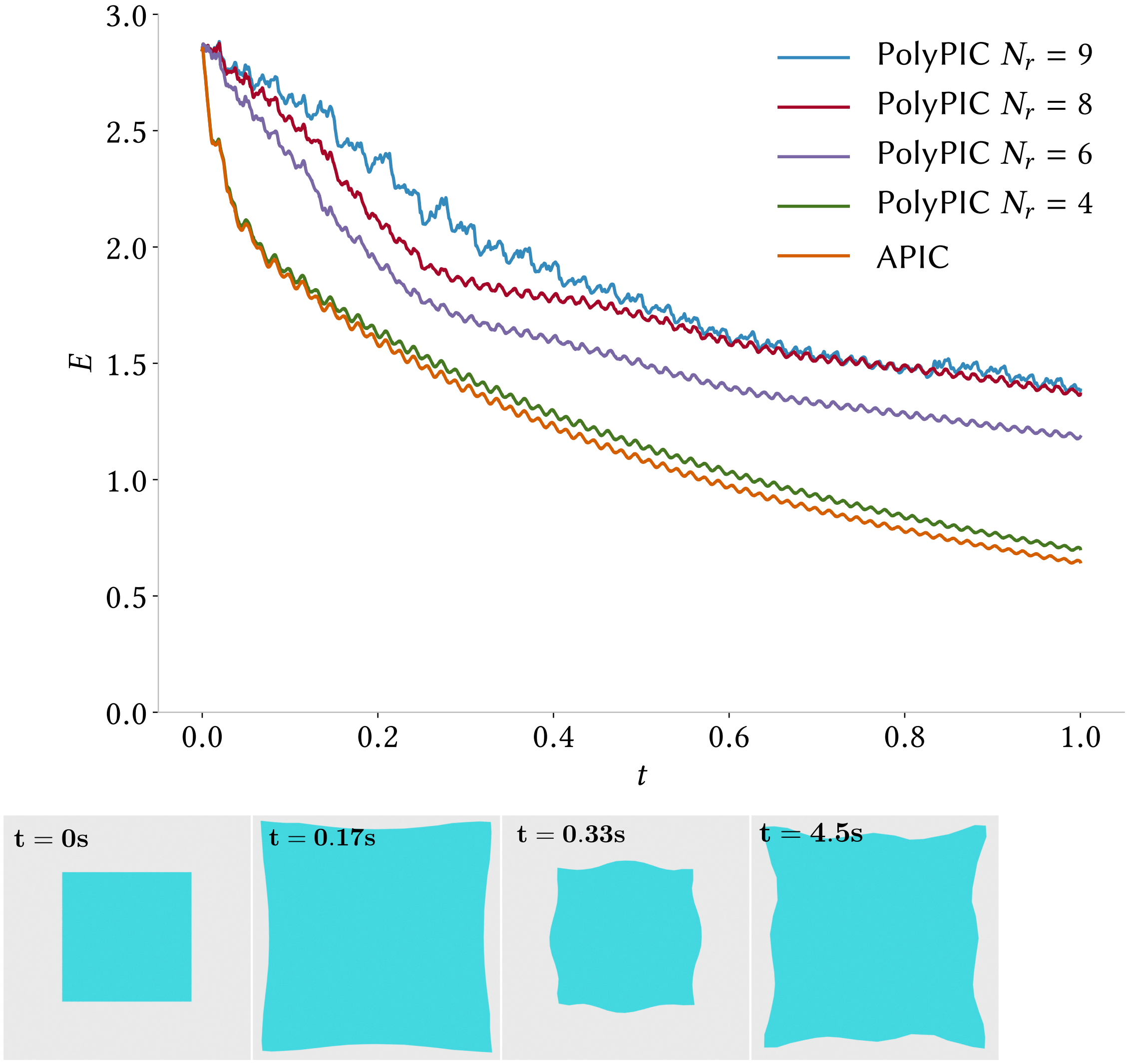

Figure 2.7: **Energy conservation**. We plot of the total energy as a function of time for an elastic square with initial compressive dilation. The energy is calculated as the sum of the elastic potential energy on the particles and the kinetic energy on the grid.

2.6.1 Incompressible flow

In Figure 2.8 we simulate a vortex sheet by setting the velocity inside a circle to be initially rotating relative to a stationary surrounding fluid. The discontinuity in the velocity induces vorticity at the interface which produces intricate flow patterns. We compare PolyPIC to FLIP and APIC and see that it better resolves the vorticial flow.

In Figure 2.9 we simulate an ink droplet in an ambient incompressible fluid by dropping liquid onto a free surface. We only render the particles in the jet. Note that the ink and water are both simulated as the same incompressible fluid. We compare PolyPIC to FLIP and APIC and see that it again better captures the transition to turbulence. We note that PolyPIC works well even when the grid resolution is rather low. Figure 2.9 was run with a relatively low grid resolution $64 \times 256 \times 64$. We used 8 simulated particles per cell, and 8000 passively advected tracer particles per cell in a post-process for visualization.

Figure 2.10 demonstrates a 3D version of the vortex sheet. The cylinder is initially rotating about its axis relative to a stationary ambient fluid. It was also run on a low resolution grid ($88 \times 132 \times 88$) with 8 simulated particles per cell for simulation and 216 passively advected tracer particles per cell in a post-process for visualization. Despite the low resolution simulation, intricate flow patterns are observed.

For all incompressible flow examples we use constant, linear and multilinear modes (i.e. $N_r = 2^d$) with PolyPIC. This is the maximum number of modes we can use because the grid interpolation in the incompressible flow solver is multilinear ($N_B = 1$) and, as discussed in Section §2.4.3.2, the number of reduced modes is bounded by $N_r \leq (N_B + 1)^d$.

2.6.2 MPM elastoplasticity

In Figure 2.4 we demonstrated the increasingly energetic nature of PolyPIC elastoplasticity simulations as we add more polynomial modes. Note that with $N_r = 6$ modes the sand flows more freely and splashes off the jello more dramatically, while the Jell-O bounces more readily.

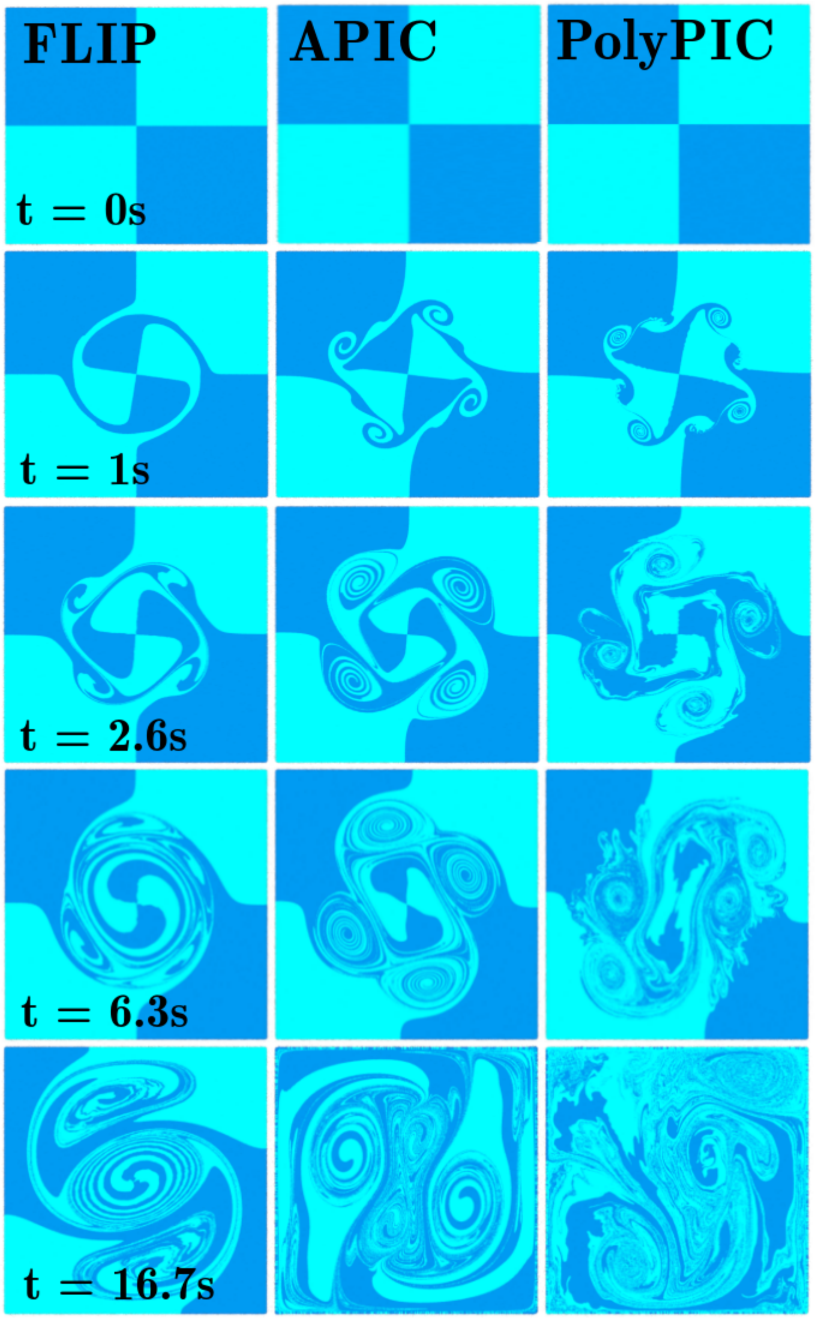

Figure 2.8: **Vortex sheet.** We compare from left to right FLIP, APIC, and PolyPIC with 2D incompressible flow. The initial conditions are of a rotating circle surrounded by stationary fluid. This creates a vortex sheet which our method effectively resolves. The bottom row shows that despite the energetic nature of our method, our simulations are stable at long runtimes.

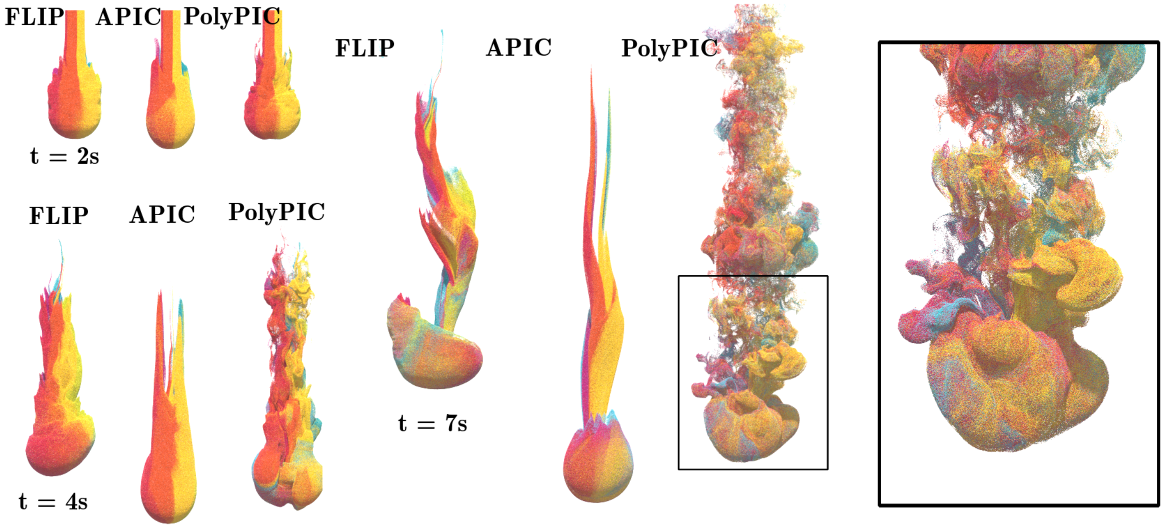

Figure 2.9: **Ink drop.** We compare from left to right FLIP, APIC, and PolyPIC for an inkjet in an ambient incompressible fluid. PolyPIC more effectively resolves the vorticial details.

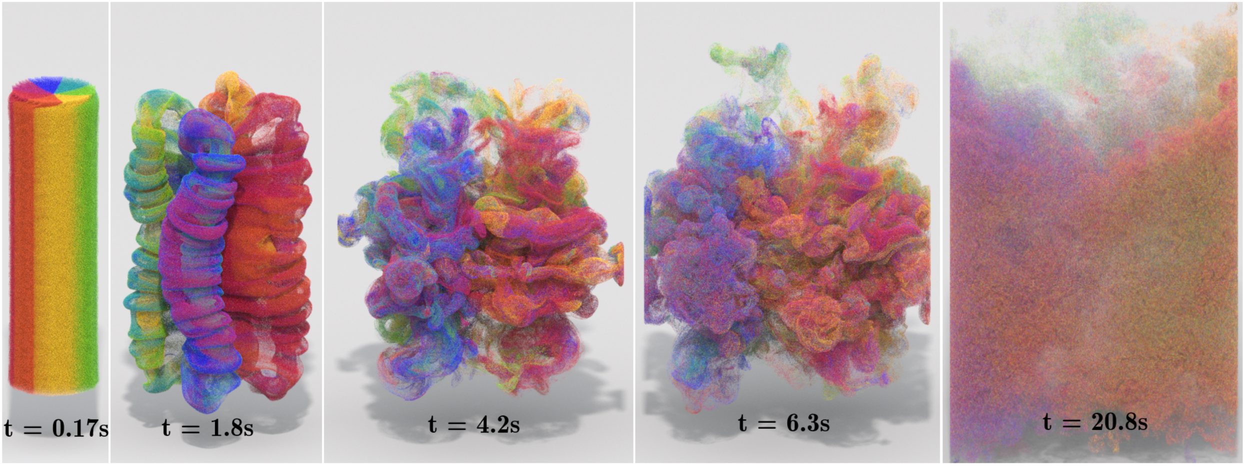

Figure 2.10: **Rotating column of colored dust**. We demonstrate intricate vorticial patterns that arise from simple initial conditions with incompressible flow. PolyPIC achieves great detail with modest spatial grid resolution (88 × 132 × 88). The rightmost image shows that despite the energetic nature of our method, our simulations are stable at long runtimes.

In Figure 2.7 we demonstrate the improved energy conservation of our method over APIC. In this scenario, a 2D hyperelastic square is initially compressed. The total energy of the system should be conserved with these initial and boundary conditions (zero traction). As we add more polynomial modes, the energy preservation improves. In Figure 2.6, we demonstrate how the increased energy retention affects the dynamics of a Jell-O cube dropped on the ground.

2.6.3 Accuracy and the number of modes

We verify that adding additional modes increases the accuracy of the simulation. In Figures 2.4 and 2.5 we examine the case of granular sand flowing from a container onto Jell-0. In Figure 2.4 we see that PolyPIC with $N_r = 4$ and APIC are less energetic than PolyPIC with $N_r = 6$. The flow of the sand in the container suffers from more numerical friction with PolyPIC $N_r = 4$ and APIC, therefore sand flows out of the container much slower. We can see this because the containers are still quite full in the final frame with PolyPIC $N_r = 4$ and APIC compared to PolyPIC with $N_r = 6$. In Figure 2.5 we rerun the same simulations but with higher grid and particle resolution. At this resolution, the PolyPIC $N_r = 4$ and APIC containers are all nearly empty in the final frame and as a result all flows are similarly energetic, indicating that PolyPIC with more modes gives a more accurate result since it is more predictive of the refined behavior.

2.6.4 Momentum conservation

We verify the angular momentum conservation properties of the PolyPIC transfers. In Figure 2.11 we plot the linear and angular momentum over the course of the time step for the falling Jell-O example shown in Figure 2.6. Even though the PolyPIC transfers conserve the momenta, the grid momentum update and the application of boundary conditions (Section §2.4.2) are not momentum conserving. To illustrate the conservation of the momentum

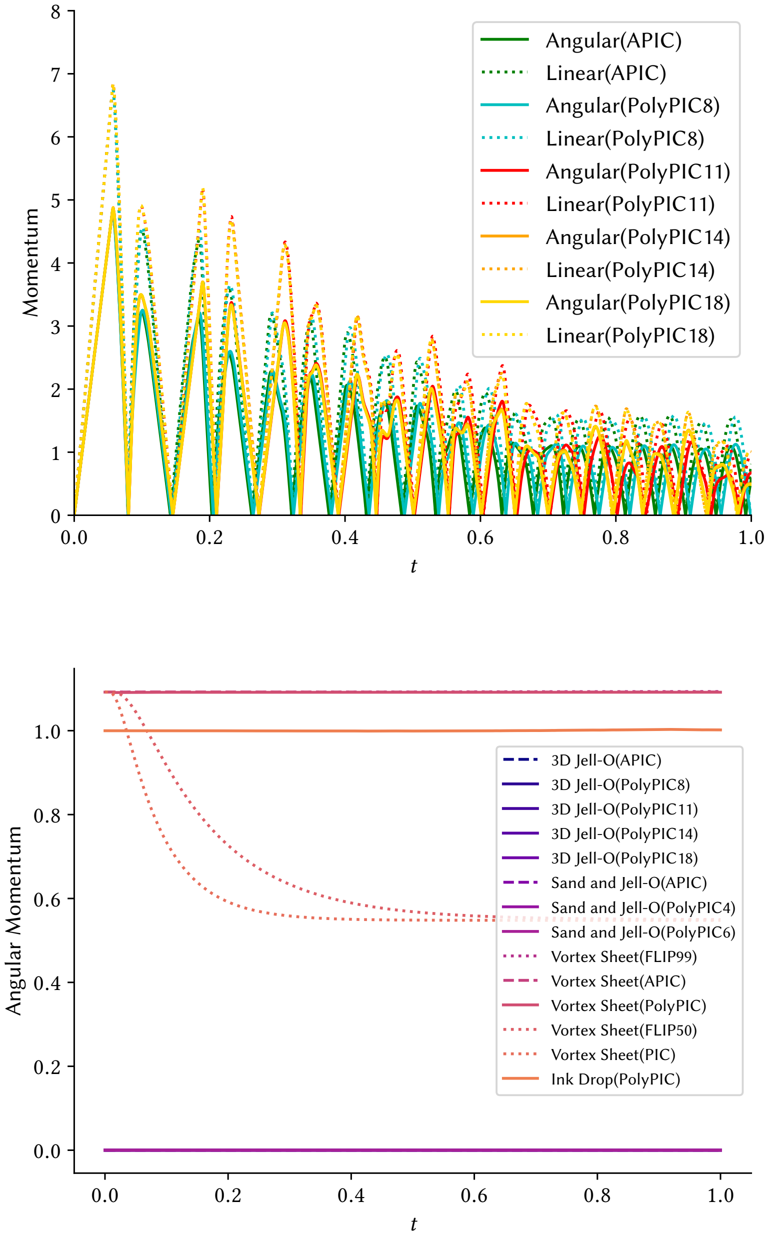

Figure 2.11: **Momentum conservation**. The top figure plots the linear and angular momenta for the falling Jell-O's in Figure 2.6. The bottom illustrates the angular momentum loss resulting form transfers. We plot the momenta $\hat{\mathbf{l}}^n$ and $\hat{\mathbf{p}}^n$ from Equation (2.17) to monitor the transfers effects on conservation. APIC and PolyPIC preserve angular momentum during transfers, however the FLIP/PIC blends are commonly used in incompressible flow simulations do not. We illustrate this by comparing with increasing amounts of PIC.

	Seconds/Frame	$\Delta t_{\max}$	Particles	Cores
Ink Drop(FLIP99)	20.569	5×10^{-2}	3.64M	16
Ink Drop(APIC)	23.188	5×10^{-2}	3.64M	16
Ink Drop(PolyPIC)	31.466	5×10^{-2}	3.64M	16
Cylinder(PolyPIC)	146.744	2×10^{-1}	7.86M	12
Vortex Sheet(FLIP99)	2.367	1×10^{-1}	0.97M	20
Vortex Sheet(APIC)	2.739	1×10^{-1}	0.97M	12
Vortex Sheet(PolyPIC)	2.760	1×10^{-1}	0.97M	20
Sand & Jello(APIC)	11.582	4×10^{-5}	59.7K	12
Sand & Jello(PolyPIC4)	12.616	4×10^{-5}	59.7K	12
Sand & Jello(PolyPIC6)	17.682	4×10^{-5}	59.7K	12
Jello(APIC)	4.882	2×10^{-4}	17.5K	48
Jello(PolyPIC8)	5.713	2×10^{-4}	17.5K	48
Jello(PolyPIC11)	5.562	2×10^{-4}	17.5K	48
Jello(PolyPIC14)	5.512	2×10^{-4}	17.5K	48
Jello(PolyPIC18)	5.852	2×10^{-4}	17.5K	48

Table 2.3: We list the time step sizes, run times, particle counts and number of cores used for our simulations. We note that the Jell-O examples demonstrate that increasing the number of reduced modes N_r in PolyPIC only moderately increases the computational cost over APIC.

in the transfers, we can monitor

$$\hat{\mathbf{l}}^n = \mathbf{l}^n_{P2G} + \sum_{m=1}^{n-1} \mathbf{l}^m_{\text{grid}} - \mathbf{l}^m_{P2G},\ \ \hat{\mathbf{p}}^n = \mathbf{p}^n_{P2G} + \sum_{m=1}^{n-1} \mathbf{p}^m_{\text{grid}} - \mathbf{p}^m_{P2G} \tag{2.17}$$

where $\mathbf{l}^n = \sum_{\mathbf{i}} \mathbf{x}_{\mathbf{i}} \times m_{\mathbf{i}} \mathbf{v}^n_{\mathbf{i}}$ and $\mathbf{p}^n = \sum_{\mathbf{i}} m_{\mathbf{i}} \mathbf{v}^n_{\mathbf{i}}$ are the angular and linear momenta on the grid. $\mathbf{l}^m_{P2G}$ $\mathbf{l}^m_{P2G}$ are computed after the transfer from particle to grid (Section §2.4.1) and $\mathbf{l}^m_{\text{grid}}$ and $\mathbf{p}^m_{\text{grid}}$ are computed after the grid momentum update (Section §2.4.2). The quantities $\mathbf{l}^m_{\text{grid}} - \mathbf{l}^m_{P2G}$ and $\mathbf{p}^m_{\text{grid}} - \mathbf{p}^m_{P2G}$ are the momenta lost during the grid momentum update at time step t^m. This is the only source of angular momentum loss for APIC and PolyPIC and thus the quantities in Equation (2.17) should be constant for those methods. We visualize the angular momentum loss from transfers in Figure 2.11. The straight lines indicate conservation.

2.7 Discussion and Limitations

While our method is a natural extension to the APIC approaches in [JSS15, JST17], it has some apparent drawbacks. Adding more polynomial modes helps to increase the energy conservation during transfers which reduces numerical dissipation. However, numerical dissipation is often desirable. Most everyday examples of elasticity involve some type of damping term and numerical dissipation is often an acceptable approximation to this. Also, numerical dissipation in the transfers can help to stabilize the method. At large time steps PIC calculations will go unstable and numerical dissipation can increase the critical time step size at which instability dominates. Indeed we found that introducing too many modes for the hyperelastic simulations can lead to small time steps. In general, we set $\Delta t = \min(\frac{\text{CFL}\Delta x}{\text{max\_}v}, \Delta t_{\text{max}})$ where max_v is the magnitude of the maximum velocity, 0¡CFL¡1 and Δt_{max} is typically about 1e-3 (see Table 2.3). If simulations go unstable, we shrink Δt_{max}. We notice that Δt_{max} will decrease to around 1e-5 if we use larger numbers of modes and this can lead to longer run times. We typically use explicit symplectic Euler (SE) integration for the grid momentum update and energy loss in transfers helps to stabilize it.

Our approach incurs storage proportionate to the number of modes since each particle must store the coefficient of the polynomial bases used to locally represent the velocity field, however in the case of affine polynomials this is equivalent to storing the velocity and velocity derivative, thus for $Nr \approx d + 1$ the storage is approximately that of original APIC. Similarly, our transfers have computational cost that is linear in the number of reduced modes. Therefore, the run time will increase slightly with additional modes. We demonstrate this by progressively adding more modes in the Jell-O examples in Figure 2.6. Run times are given in Table 2.3.

The number of nodes with nonzero weights implies a threshold on the number of velocity modes $N_r \leq (N_B + 1)^d$. For larger N_r the system for $\mathbf{c}_p^{n+1}$ is overdetermined since there would be more modes than grid node velocity values to determine them from. In principle, our method would still work however we did not investigate this possibility. Interestingly, we noticed that with $N_r = (N_B + 1)^d$ the transfer from grid to particle then back to grid is lossless (neglecting motion of the particles) (see Appendix §A.3).

CHAPTER 3

Numerical Simulation for Thin Shell with Frictional Contact

3.1 Mathematical Details and Notation

We use bold face (e.g. $\mathbf{v}$) to denote vector and tensor quantities and plain text (e.g. v) to denote scalar quantities. We use brackets around bold face to denote matrices associated with a tensor in a given basis (e.g $[\mathbf{M}] \in \mathbb{R}^{3\times 3}$ is the matrix of entries $m_{ij} \in \mathbb{R}$ where tensor $\mathbf{m} = m_{ij}\mathbf{e}_i \otimes \mathbf{e}_j$). We use the convention that Greek indices (e.g. a_α) range from $1-2$ and Latin indices (e.g. b_i) range from $1-3$. We use hat notation to indicate the upper left 2×2 sub matrix of a given matrix (e.g. $\left[\hat{\mathbf{M}}\right] \in \mathbb{R}^{2\times 2}$ consists of entries $m_{\alpha\beta}$ from $[\mathbf{M}] \in \mathbb{R}^{3\times 3}$). Unless otherwise stated, we use the summation convention for repeated indices. For a set of (covariant) basis vectors $\mathbf{v}_i$, we use $\mathbf{v}^j$ to denote the corresponding contravariant basis vectors satisfying $\mathbf{v}_i \cdot \mathbf{v}^j = \delta_i^j$. $|\cdot|$ is used to denote the L_2 norm of a vector.

We assume shells have constant thickness τ and use $\omega^\tau = \omega \times [-\frac{\tau}{2}, \frac{\tau}{2}]$ to parameterize the domain of the shell where ω is two-dimensional parameter domain for the mid-surface of the shell. We use $\bar{\mathbf{x}} : \omega \to \bar{\Omega}$ and $\mathbf{x} : \omega \to \Omega_t$ to denote the mappings from the mid-surface parameter domain to the reference ($\bar{\Omega}$) and time t (Ω_t) configurations of the mid-surface. Similarly we use $\bar{\mathbf{r}} : \omega^\tau \to \bar{\Omega}^\tau$ and $\mathbf{r} : \omega^\tau \to \Omega_t^\tau$ to denote mappings from the shell parameter domain to the reference ($\bar{\Omega}^\tau$) and time t (Ω_t^τ) configurations of the shell. We illustrate this in Figure 3.1. We will use $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3) \in \omega^\tau$ to denote coordinates in parameter space. We refer to surfaces $\mathbf{s}(\xi_1, \xi_2) = \mathbf{r}(\xi_1, \xi_2, \hat{\xi_3})$ in the shell with fixed values of the thickness parameter $\hat{\xi_3}$ as laminae and we refer to lines in the $\mathbf{l}(\xi_3) = \mathbf{r}(\hat{\xi_1}, \hat{\xi_2}, \xi_3)$ with fixed values of the surface parameters $\hat{\xi_1}, \hat{\xi_2}$ as fibers. We illustrate fibers and laminae in Figure 3.3.

3.2 Shell Kinematics

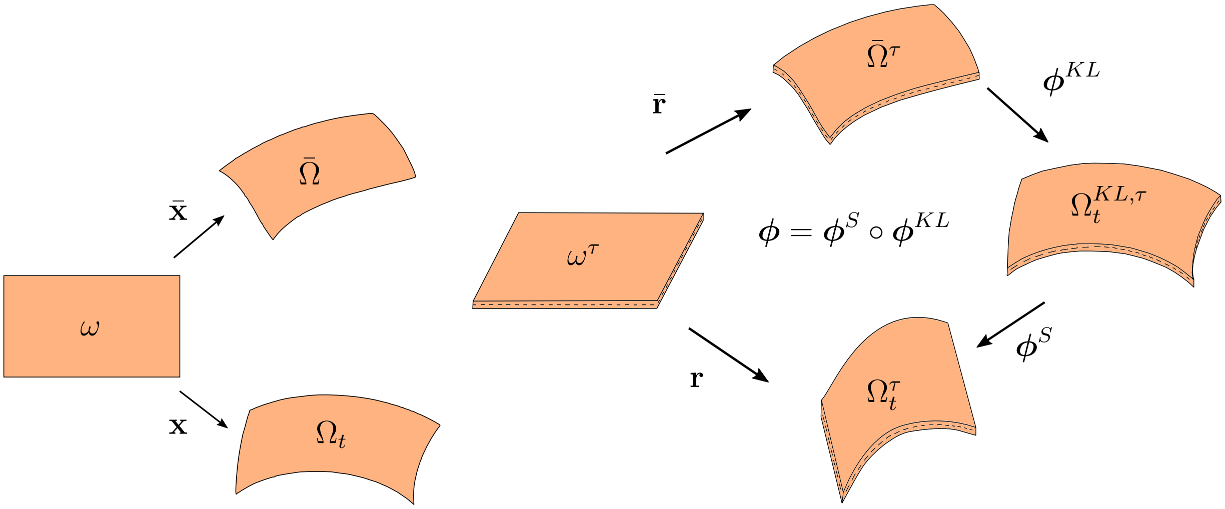

Figure 3.1: **Shell Kinematics**. On the left, the mid-surface mappings are illustrated, and on the right the corresponding volumetric shell mappings are shown.

We assume the kinematics of a continuum shell

$$\bar{\mathbf{r}}(\boldsymbol{\xi}) = \bar{\mathbf{x}}(\xi_1, \xi_2) + \xi_3 \bar{\mathbf{a}}_3(\xi_1, \xi_2), \;\; \mathbf{r}(\boldsymbol{\xi}) = \mathbf{x}(\xi_1, \xi_2) + \xi_3 \mathbf{a}_3(\xi_1, \xi_2) \tag{3.1}$$

where $\bar{\mathbf{a}}_3$ is the unit normal to the mid-surface and $\mathbf{a}_3$ is the stretched and sheared image of $\bar{\mathbf{a}}_3$ under the motion of the shell. We use $\bar{\mathbf{a}}_\alpha = \frac{\partial \bar{\mathbf{x}}}{\partial \xi_\alpha}$ to denote the tangents to the mid-surface of the reference shell. When combined with $\bar{\mathbf{a}}_3 = \frac{\bar{\mathbf{a}}_1 \times \bar{\mathbf{a}}_2}{|\bar{\mathbf{a}}_1 \times \bar{\mathbf{a}}_2|}$, they form a complete basis for $\mathbb{R}^3$ (see Figure 3.3).

We decompose the motion of the shell into two steps

$$\mathbf{r}(\boldsymbol{\xi}) = \boldsymbol{\phi}^S(\mathbf{r}^{KL}(\boldsymbol{\xi})). \tag{3.2}$$

The first step $\mathbf{r}^{KL} : \omega^\tau \rightarrow \Omega_t^{KL,\tau}$ does not see shearing or compression normal to the mid-surface. That is, lines originally normal to the midsurface rotate and translate with the midsurface so that they remain constant length and normal to the midsurface. This is

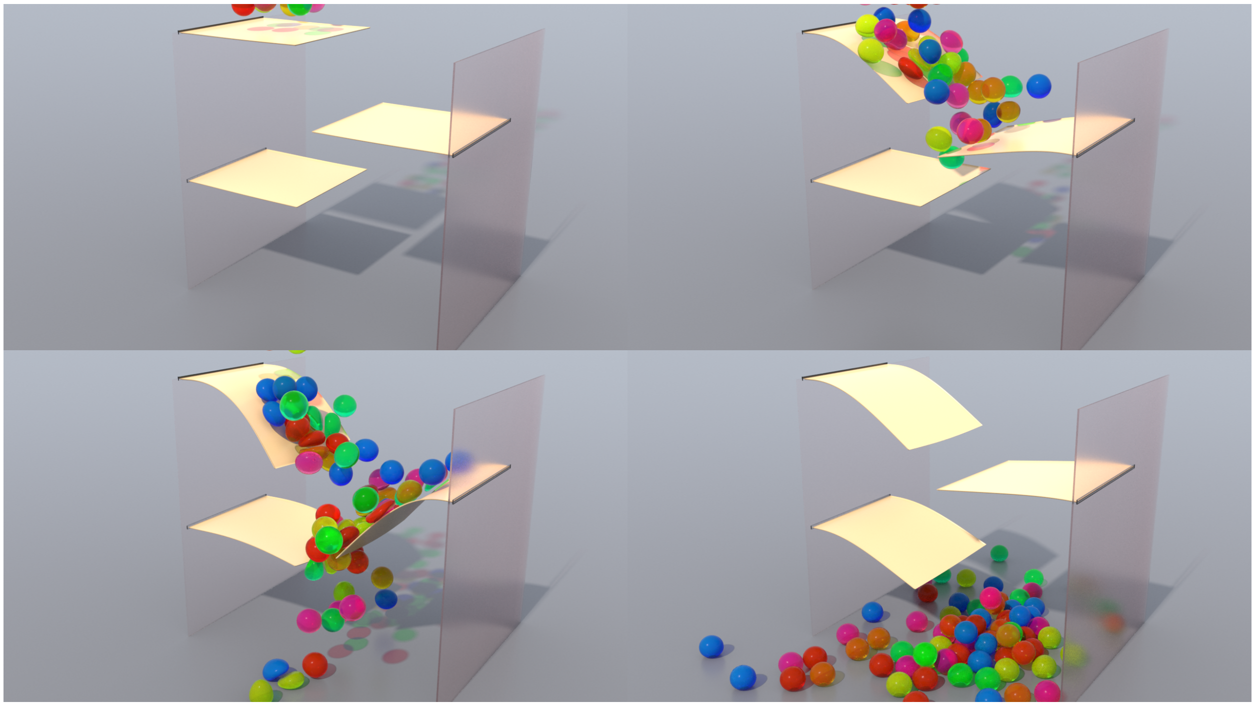

Figure 3.2: **Elastic spheres on diving boards**. We demonstrate appealing dynamics achieved with self-collision and appreciable bending for shells. Both the spheres and the diving boards are simulated as thin shells.

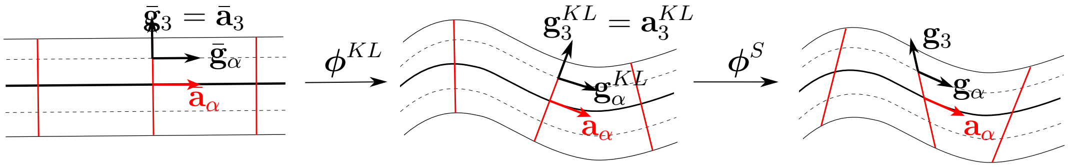

Figure 3.3: **Continuum shell/Kirchhoff-Love splitting**. Mid-surface tangents and fibers are shown in red. Laminae are shown as dashed curves, and the local frame at a point on a lamina is shown in black. On the left is the undeformed reference configuration, while the deformed configuration is on the right, and the middle shows the intermediate Kirchhoff-Love deformation.

consistent with a Kirchhoff-Love kinematic assumption

$$\mathbf{r}^{KL}(\boldsymbol{\xi}) = \mathbf{x}(\xi_1, \xi_2) + \xi_3 \mathbf{a}_3^{KL}(\xi_1, \xi_2). \tag{3.3}$$

Here $\mathbf{a}_3^{KL}$ is the unit normal to the mid-surface which satifsies $\mathbf{a}_3^{KL} = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|}$ where $\mathbf{a}_\alpha = \frac{\partial \mathbf{x}}{\partial \xi_\alpha}$. The second step $\boldsymbol{\phi}^S : \Omega_t^{KL,\tau} \to \Omega_t^\tau$ does not move the mid-surface but captures the shearing and compression/extension of material normal to the mid-surface. That is, lines that remained normal to the midsurface and with constant length in the Kirchhoff-Love mapping $\mathbf{r}^{KL}$ are allowed to change length and shear under the mapping $\boldsymbol{\phi}^S$, thus becoming non-normal to the midsurface in general (see Figure 3.3).

3.2.1 Deformation gradient

The motion of the shell from the reference configuration to the time t configuration is then obtained from the composition $\boldsymbol{\phi} : \bar{\Omega}^\tau \to \Omega_t^\tau$, $\boldsymbol{\phi}(\mathbf{X}) = \mathbf{r}(\bar{\mathbf{r}}^{-1}(\mathbf{X}))$ for $\mathbf{X} \in \bar{\Omega}^\tau$. The elastic and frictional contact responses of our model are characterized in terms of the spatial derivative (our deformation gradient) of this mapping. The deformation gradient of the motion is $\mathbf{F} = \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{X}} = \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}} \left(\frac{\partial \bar{\mathbf{r}}}{\partial \boldsymbol{\xi}} \right)^{-1}$, which can further be expressed in terms of derivatives from the parameter space $\mathbf{g}_i = \frac{\partial \mathbf{r}}{\partial \xi_i}$ and $\bar{\mathbf{g}}_i = \frac{\partial \bar{\mathbf{r}}}{\partial \xi_i}$ as $\mathbf{F} = \mathbf{g}_i \otimes \bar{\mathbf{g}}^i$. Here $\bar{\mathbf{g}}^i$ are the contravariant basis vectors associated with $\bar{\mathbf{g}}_i$. Furthermore, the composition of motion in Equation (3.2) leads to the multiplicative decomposition

$$\mathbf{F} = \mathbf{F}^S \mathbf{F}^{KL}, \quad \mathbf{F}^S = \mathbf{g}_i \otimes \mathbf{g}^{KL,i}, \quad \mathbf{F}^{KL} = \mathbf{g}_i^{KL} \otimes \bar{\mathbf{g}}^i \tag{3.4}$$

where $\mathbf{g}_i^{KL} = \frac{\partial \mathbf{r}^{KL}}{\partial \xi_i}$ and $\mathbf{g}^{KL,j}$ form the corresponding contravariant basis. We note that the third contravariant counterparts to the Kirchhoff-Love and material configuration bases are the same as their covariant counterparts because of the perservation of midsurface normals in these mappings. That is, $\mathbf{g}_3^{KL} = \mathbf{g}^{KL,3} = \mathbf{a}_3^{KL}$ and $\bar{\mathbf{g}}_3 = \bar{\mathbf{g}}^3 = \bar{\mathbf{a}}_3$ since $\mathbf{g}_\alpha^{KL} \cdot \mathbf{g}_3^{KL} = 0$ and $\bar{\mathbf{g}}_\alpha \cdot \bar{\mathbf{g}}_3 = 0$ (see [Cly17] for details).

3.2.2 Plasticity

As in Jiang et al.[JGT17], we use an elastoplastic decomposition of the motion to resolve frictional contact. Following that approach, we allow for plastic deformation in the fiber directions to enable material separation and frictional sliding. However, in order to decouple the frictional contact stress from the bending stress, we only apply the frictional contact elastoplastic decomposition to the shearing component of the motion. Furthermore, unlike in Jiang et al.[JGT17] we also allow for plastic deformation in the laminae to account for yielding and denting of the shell. This plastic decomposition is applied to the motion in the Kirchhoff-Love component of the motion.

The frictional contact elastic stress model in Jiang et al. [JGT17] penalizes compression and shearing of the surface normals. Since the Kirchhoff-Love component of the motion does not see any sliding or compression relative to the mid-surface, it is not capable of resolving frictional contact in this manner. We therefore apply this model to the shearing and compression/extension component of the shearing motion $\mathbf{F}^S = \mathbf{F}^{S,E}\mathbf{F}^{S,P}$ as

$$\mathbf{F}^{S,E} = \mathbf{g}_\alpha \otimes \mathbf{g}^{KL,\alpha} + \mathbf{a}_3^E \otimes \mathbf{g}_3^{KL}, \tag{3.5}$$

$$\mathbf{F}^{S,P} = \mathbf{g}_\alpha^{KL} \otimes \mathbf{g}^{KL,\alpha} + \mathbf{a}_3^P \otimes \mathbf{g}_3^{KL}. \tag{3.6}$$

Here $\mathbf{a}_3^E$ represents the shearing and compression/extension of normals in the shell that is penalized elastically. Coulomb friction constrains how much shearing and compression is penalized. $\mathbf{a}_3^P$ is the discarded shearing and extension in the fiber direction from plastic yielding associated with this constraint. They are related through $\mathbf{F}^{S,E}\mathbf{a}_3^P = \mathbf{a}_3$. We note $\mathbf{F}^{S,P}$ does not affect components in the laminae since we do not want the frictional contact response to couple with the elastoplasticty of the Kirchhoff-Love component of the shell motion.

To allow for yielding and denting of the shell in response to loading, we decompose the Kirchhoff-Love component of the motion into lamina elastic and plastic parts $\mathbf{F}^{KL} = \mathbf{F}^{KL,E}\mathbf{F}^{KL,P}$

$$\mathbf{F}^{KL,E} = \mathbf{g}_\alpha^{KL} \otimes \mathbf{g}^{P,\alpha} + \mathbf{g}_3^{KL} \otimes \bar{\mathbf{g}}_3, \tag{3.7}$$

$$\mathbf{F}^{KL,P} = \mathbf{g}_\alpha^P \otimes \bar{\mathbf{g}}^\alpha + \bar{\mathbf{g}}_3 \otimes \bar{\mathbf{g}}_3 \tag{3.8}$$

Here the form of $\mathbf{F}^{KL,P}$ is designed to not affect the motion normal to the mid-surface since the elastoplasiticty of denting and wrinkling is expressed only in terms of the lamina components of defomraiton. The exprssion for $\mathbf{F}^{KL,E}$ is then what remains to satisfy the constraint $\mathbf{F}^{KL} = \mathbf{F}^{KL,E}\mathbf{F}^{KL,P}$. We note that the $\mathbf{g}_\alpha^P$ (with $\mathbf{g}_\alpha^P \cdot \bar{\mathbf{g}}_3 = 0$) in Equation (3.8) for

$\mathbf{F}^{KL,P}$ express the forgotten deformation of plastic yielding in the lamina that is associated with denting and wrinkling. The $\{\mathbf{g}^{P\alpha}, \bar{\mathbf{g}}_3\}$ are the contravariant counterparts to $\{\mathbf{g}^P_\alpha, \bar{\mathbf{g}}_3\}$. Lastly, $\bar{\mathbf{g}}_3$ is the same in the covariant and contravariant bases as in Equation (3.4).

3.3 Elastic Stress and Plastic Constraints

We define our elastoplastic constitutive response to deformation and frictional contact terms of potential energy in the shell. We decompose the total elastic potential as a sum of contributions from the Kirchhoff-Love (lamina elasticity, denting wrinkling etc.) and shearing (frictional contact) potentials. The contribution from the Kirchhoff-Love motion is

$$\Psi^{KL} = \int_\omega \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \psi(\mathbf{F}^{KL,E}) \left| \frac{\partial \bar{\mathbf{r}}}{\partial \xi} \right| d\boldsymbol{\xi}. \tag{3.9}$$

and the total elastic potential energy of the shell is

$$\Psi^{CS} = \Psi^{KL} + \int_\omega \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \chi(\mathbf{F}^{S,E}) \left| \frac{\partial \bar{\mathbf{r}}}{\partial \xi} \right| d\boldsymbol{\xi} \tag{3.10}$$

where $\psi(\mathbf{F}^{KL,E})$ is the elastic potential energy density of the Kirchhoff-Love motion and $\chi(\mathbf{F}^{S,E})$ is the energy density of the normal shearing and compression in the continuum shell motion.

These potentials are defined from energy densities $\psi(\mathbf{F}^{KL,E})$ and $\chi(\mathbf{F}^{S,E})$ respectively. In general, a potential energy density Ξ of this type is related to the material Kirchhoff stress $\boldsymbol{\tau}$ through $\boldsymbol{\tau} = \frac{\partial \Xi(\mathbf{F}^E)}{\partial \mathbf{F}^E} \mathbf{F}^E$. It is the stress defined through this relation that will directly affect our MPM implementation. In our elastoplastic model, the stress must satisfy certain constraints related to bending and denting as well as frictional contact. In the sections that follow we define these elastic stresses and their associated plastic constraints.

3.3.1 Bending and lamina potential

The energy density $\psi(\mathbf{F}^{KL,E})$ penalizes only the deformation in the laminae (zero transverse normal stress) since the Kirchhoff-Love kinematics preclude shearing and compression of the fibers. The stress in the material is the derivative of this potential with respect to strain (see Appendix §B.1 for derivation). Our approach supports any potential used in Kirchhoff-Love shell models. In particular we use the orthotropic model for woven fabrics from Clyde et al.[CTT17] in Figures 3.10a and 3.10b. Here we provide the derivation of a simple energy density useful for applications with denting that is isotropic in the lamina directions while satisfying the zero transverse normal stress condition.

With Kirchhoff-Love kinematics, the lamina directions $\bar{\mathbf{g}}_\alpha = \bar{\mathbf{a}}_\alpha + \xi_3 \bar{\mathbf{a}}_{3,\alpha}$ and $\mathbf{g}^{KL}_\alpha = \mathbf{a}_\alpha + \xi_3 \mathbf{a}^{KL}_{3,\alpha}$ are always tangent to the mid-surface since $\bar{\mathbf{g}}_\alpha \cdot \bar{\mathbf{a}}_3 = \mathbf{g}^{KL}_\alpha \cdot \mathbf{a}^{KL}_3 = 0$. In order to satisfy the zero transverse normal stress conditions, we design a potential density with respect to the lamina directions by first writing the Kirchhoff-Love deformation in the reference mid-surface lamina/fiber basis $\mathbf{F}^{KL,E} = F^{KL,E}_{ij} \bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j$. Here the directions $\bar{\mathbf{a}}_\alpha$ are the tangents to the midsurface in the reference configuration and $\bar{\mathbf{a}}_3$ is the normal. This choice of basis more clearly identifies deformations in the laminae and normal directions since $F^{KL,E}_{\alpha\beta}$ are then components of deformation in the laminae. The right Cauchy-Green strain is $\mathbf{C} = C_{ij} \bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j$ with $C_{ij} = F^{KL,E}_{ki} F^{KL,E}_{kj}$. We define the matrix $[\hat{\mathbf{C}}] \in \mathbb{R}^{2\times 2}$ with entries $C_{\alpha\beta}$. This is the upper left 2×2 block of the matrix of C_{ij} entries and it represents strain in the lamina. We use a model that is quadratic in the right Hencky strain $\left[\boldsymbol{\epsilon}^R\right] = \frac{1}{2}\log([\hat{\mathbf{C}}])$

$$\psi(\mathbf{F}^{KL,E}) = \mu \epsilon^R_{\alpha\beta} \epsilon^R_{\alpha\beta} + \frac{\lambda}{2}(\epsilon^R_{\delta\delta})^2. \tag{3.11}$$

Here the $\epsilon^R_{\alpha\beta}$ are the entries in $\left[\boldsymbol{\epsilon}^R\right] \in \mathbb{R}^{2\times 2}$ and μ, λ are Lame parameters that can be set intuitively from Young's modulus and Poisson ratio to control stiffness and incompressibility in the lamina. We choose the quadratic in Hencky strain model because it simplifies the return mapping during plastic yielding (see Section §3.3.2).

It is convenient for our MPM implementation as well as for the plasticity constraints to work with the Kirchhoff stress $\boldsymbol{\tau}$. It is related to the more commonly used Cauchy stress $\boldsymbol{\sigma}$

as $\boldsymbol{\tau} = \det(F)\boldsymbol{\sigma}$. The derivation of the Kirchhoff stress in terms of the potential is given in Appendix §B.1. We summarize the expression as

$$\boldsymbol{\tau}^{KL} = \tau_{\alpha\beta}\mathbf{q}_\alpha^{KL,E} \otimes \mathbf{q}_\beta^{KL,E}, \quad \tau_{\alpha\beta}^{KL} = 2\mu\epsilon_{\alpha\beta}^L + \lambda\epsilon_{\gamma\gamma}^L\delta_{\alpha\beta}. \tag{3.12}$$

Here we write the stress in terms of the basis defined by the directions $\mathbf{q}_i^{KL,E}$ obtained from the QR decomposition $\mathbf{F}^{KL,E} = r_{ij}^{KL,E}\mathbf{q}_i^{KL,E} \otimes \bar{\mathbf{a}}_j$ with respect to the reference lamina/fiber basis $\bar{\mathbf{a}}_j$. Since the Kirchhoff-Love component of the motion preserves normals to the mid-surface, the first two directions $\mathbf{q}_\alpha^{KL,E}$ are tangent to the deformed lamina and the third direction $\mathbf{q}_3^{KL,E}$ is normal to the mid-surface. Therefore, since $\boldsymbol{\tau}^{KL}$ is expressed only in terms of $\mathbf{q}_\alpha^{KL,E}$, we see that it satisfies the zero transverse normal stress condition since it has no components in the directions normal to the laminae. We use $\epsilon_{\alpha\beta}^L$ to denote the entries in the left Hencky strain matrix $[\boldsymbol{\epsilon}^L] = \frac{1}{2}\log([\hat{\mathbf{r}}^{KL,E}][\hat{\mathbf{r}}^{KL,E}]^T) \in \mathbb{R}^{2\times 2}$. Here, $[\hat{\mathbf{r}}^{KL,E}] \in \mathbb{R}^{2\times 2}$ is the matrix with entries $r_{\alpha\beta}^{KL,E}$. These are the components of the deformation gradient $\mathbf{F}^{KL,E}$ related to the lamina strain. This formula follows directly from the definition of the energy in Equation (3.11) and we provide details of the derivation in Appendix §B.1.

3.3.2 Denting yield condition and return mapping

In order to produce permanent denting and wrinkling phenomena resulting from excessive straining, we introduce a notion of yield stress. Intuitively, stresses satisfying the yield stress criteria are those associated with elastic, non-permanent deformation in the shell. Those that do not satisfy the condition are non-physical and permanent plastic deformation will occur to prevent them from happening. We apply the von Mises yield condition to the Kirchhoff-Stress in Equation (3.12). This condition states that the shear stress (or magnitude of the deviatoric component of the stress) must be less than a threshold c_{vM} before permanent plastic deformation occurs

$$f_{vM}(\boldsymbol{\tau}) = |\boldsymbol{\tau} - \frac{\operatorname{tr}(\boldsymbol{\tau})}{3}\mathbf{I}|_F \leq c_{vM}. \tag{3.13}$$

This condition defines a cylindrical region of feasible states in the principal stress space since

$$f_{vM}(\boldsymbol{\tau}) = \sqrt{\frac{2}{3}\left({\tau_1}^2 + {\tau_2}^2 + {\tau_3}^2 - (\tau_1\tau_2 + \tau_2\tau_3 + \tau_1\tau_3)\right)} \tag{3.14}$$

where $\boldsymbol{\tau} = \sum_i \tau_i \mathbf{u}_i \otimes \mathbf{u}_i$ with principal stresses τ_i. Stresses with principal values in the cylinder do not produce any permanent deformation. Note that zero stress is inside the cylinder. As deformation becomes significant enough that the principal stresses reach the boundary of the cylinder, permanent plastic denting and wrinkling will occur. The zero transverse normal stress nature of $\boldsymbol{\tau}^{KL} = \sum_\alpha \tau_\alpha^{KL} \mathbf{u}_\alpha \otimes \mathbf{u}_\alpha$ means that its principal stresses are always in a plane and thus the feasible region is ellipsoidal intersection of the cylinder and the plane (see Appendix §B.7 for illustration).

In practice, the yield condition is satisfied via projection (or return mapping) of the stress to the feasible region. During simulation, we first take a time step to create a trial state of stress ignoring the yield condition. By ignoring the condition, we essentially assume the material undergoes no further plastic deformation. We use $\mathbf{F}^{KL,E^{\mathrm{tr}}}$, $\mathbf{F}^{KL,P^{\mathrm{tr}}}$ to denote this trial state of elastoplastic strains with associated trial stress $\boldsymbol{\tau}^{KL^{\mathrm{tr}}}$. This stress may or may not satisfy the yield condition. The trial stress $\boldsymbol{\tau}^{KL^{\mathrm{tr}}}$ is then projected to the feasible region to create $\boldsymbol{\tau}^{KL}$ which satisfies the yield condition. The elastic and plastic strains are then computed from the projected stress. We use $\mathbf{F}^{KL,E}, \mathbf{F}^{KL,P}$ to denote final elastic and plastic deformation associated with the projected stress $\boldsymbol{\tau}^{KL}$. The product of the projected elastic and plastic deformation gradients must be equal to the original deformation gradient, creating a constraint on the return mapping

$$\mathbf{F}^{KL} = \mathbf{F}^{KL,E^{\mathrm{tr}}}\mathbf{F}^{KL,P^{\mathrm{tr}}} = \mathbf{F}^{KL,E}\mathbf{F}^{KL,P}. \tag{3.15}$$

We describe the process as $\mathbf{F}^{KL,E^{\mathrm{tr}}}, \mathbf{F}^{KL,P^{\mathrm{tr}}} \to \mathbf{F}^{KL,E}, \mathbf{F}^{KL,P}$.

The projection is naturally done in terms of the QR decomposition of the trial elastic deformation gradient $\mathbf{F}^{KL,E^{\mathrm{tr}}} = r_{\alpha\beta}^{KL,E^{\mathrm{tr}}} \mathbf{q}_\alpha^{KL,E} \otimes \bar{\mathbf{a}}_\beta + \mathbf{q}_3^{KL,E} \otimes \bar{\mathbf{a}}_3$. The trial principle stresses are

$$\tau_1^{KL,\mathrm{tr}} = (2\mu + \lambda)\log(\sigma_1^{E\mathrm{tr}}) + \lambda \log(\sigma_2^{E\mathrm{tr}}) \tag{3.16}$$

$$\tau_2^{KL,\mathrm{tr}} = (2\mu + \lambda)\log(\sigma_2^{E\mathrm{tr}}) + \lambda \log(\sigma_1^{E\mathrm{tr}}) \tag{3.17}$$

where $\sigma_\alpha^{E\mathrm{tr}}$ are the singular values of the matrix $[\hat{\mathbf{r}}^{KL,E\mathrm{tr}}] \in \mathbb{R}^{2\times 2}$ with entries $r_{\alpha\beta}^{KL,E\mathrm{tr}}$ from the QR decomposition

$$[\hat{\mathbf{r}}^{KL,E\mathrm{tr}}] = [\mathbf{U}^E] \begin{pmatrix} \sigma_1^{E^{\mathrm{tr}}} & \\ & \sigma_2^{E^{\mathrm{tr}}} \end{pmatrix} [\mathbf{V}^E]^T. \tag{3.18}$$

We project the trial $\tau_\alpha^{KL,\mathrm{tr}}$ to the intersection of the von Mises yield surface and the $(1,2)$ plane to obtain the projected τ_α^{KL} from which

$$\begin{pmatrix} \log(\sigma_1^E) \\ \log(\sigma_2^E) \end{pmatrix} = \begin{pmatrix} 2\mu + \lambda & \lambda \\ \lambda & 2\mu + \lambda \end{pmatrix}^{-1} \begin{pmatrix} \tau^{KL}{}_1 \\ \tau^{KL}{}_2 \end{pmatrix}. \tag{3.19}$$

We then express the deformation gradient associated with this stress projection as $\mathbf{F}^{KL,E} = F_{\alpha\beta}^{KL,E}\mathbf{q}_\alpha^{KL,E} \otimes \bar{\mathbf{a}}_\beta + \mathbf{q}_3^{KL,E} \otimes \bar{\mathbf{a}}_3$ where $F_{\alpha\beta}^{KL,E}$ are the components of the elastic deformation gradient

$$[\hat{\mathbf{F}}^{KL,E}] = [\mathbf{U}^E] \begin{pmatrix} \sigma_1^E & \\ & \sigma_2^E \end{pmatrix} [\mathbf{V}^E]^T. \tag{3.20}$$

The projected plastic deformation gradient is computed from $\mathbf{F}^{KL,P} = \mathbf{F}^{KL,E^{-1}}\mathbf{F}^{KL}$ in order to maintain the constraint in Equation (3.15). We provide more detail in this derivation in Appendix §B.7.

3.3.2.1 Associativity and Hencky strain

The projection of the trial stress to the feasible region is done using a generalized notion of closest point. This generalized projection is derived from the associative plastic flow assumption. Associativity requires that the closest points to the feasible stress region are not traced back along lines normal to the boundary, but rather along lines parallel to a matrix times the normal [BW08]. This matrix is associated with the linearization of the constitutive model and in general it varies along the boundary. However, with the quadratic in Hencky strain model given in Equation (3.11), the matrix is constant along the boundary of the feasible region, which greatly simplifies the process of finding the generalized closest point. We illustrate this further in Appendix §B.7.

3.3.3 Frictional contact potential

As in Jiang et al.[JGT17], we resolve collision and contact through the continuum. We design the potential energy density $\chi(\mathbf{F}^{S,E})$ to penalize compression and shearing in the direction normal to the mid-surface as in Jiang et al.[JGT17]. The deformation of the fiber from the Kirchhoff-Love configuration is given by $\mathbf{a}_3^E = \mathbf{F}^{S,E}\mathbf{a}_3^{KL}$. We decompose this into shear ($\mathbf{a}_{3S}^E$) and normal ($s_3^E\mathbf{a}_3^{KL}$) components $\mathbf{a}_3^E = \mathbf{a}_{3S}^E + s_3^E\mathbf{a}_3^{KL}$ where $s_3^E = \mathbf{a}_3^E \cdot \mathbf{a}_3^{KL}$. As material normal to the cloth is compressed, the normal component s_3^E will decrease and as the material separates, it will increase. Similarly, as material slides tangentially to the shell $|\mathbf{a}_{3S}^E|$ will increase. We therefore write our potential as

$$\chi(\mathbf{F}^{S,E}) = \frac{\gamma}{2}|\mathbf{a}_{3S}^E|^2 + f(s_3^E) \tag{3.21}$$

where γ represents the amount of shear resistance and

$$f(s_3^E) = \begin{cases} \frac{k^c}{3}(1 - s_3^E)^3 & 0 \le s_3^E \le 1 \\ 0 & s_3^E > 1 \end{cases} \tag{3.22}$$

represents the resistance to compression/contact which increases with the parameter $k^c > 0$. This potential is designed to increase, and thus penalize, increasing compressive contact and shear. Note that in the case of fiber elongation ($s_3^E > 1$) there is no elastic penalty as this would be associated with cohesive contact.

The potential in Equation (3.21) is constant in the fiber direction since $\mathbf{a}_3^{KL}$ is constant along the fiber from the continuum shell kinematics. Therefore it is convenient to express the contact potential χ at all points in the fibers in terms of their values at the mid-surface $\chi(\mathbf{F}^{S,E}(\xi_1,\xi_2,\xi_3)) = \chi(\mathbf{F}^{S,E}(\xi_1,\xi_2,0))$ since

$$\int_\omega \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \chi(\mathbf{F}^{S,E}) \left| \frac{\partial \bar{\mathbf{r}}}{\partial \xi} \right| d\boldsymbol{\xi} = \int_\omega \chi(\mathbf{F}^{S,E}) \int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \left| \frac{\partial \bar{\mathbf{r}}}{\partial \xi} \right| d\boldsymbol{\xi} \tag{3.23}$$

in Equation (3.10). On the mid-surface $\mathbf{F}^{S,E}(\xi_1,\xi_2,0) = \mathbf{a}_\alpha \otimes \mathbf{a}^{KL,\alpha} + \mathbf{a}_3^E \otimes \mathbf{a}_3^{KL}$. Furthermore, since the potential varies with the normal and tangential components of $\mathbf{a}_3^E$, it is equivalent to write the energy as a function of the tensor $\mathbf{a}_\alpha \otimes \bar{\mathbf{a}}_\alpha + \mathbf{a}_3^E \otimes \bar{\mathbf{a}}_3$ since its QR decomposition with respect to the $\bar{\mathbf{a}}_i$ basis satisfies

$$\mathbf{a}_\alpha \otimes \bar{\mathbf{a}}_\alpha + \mathbf{a}_3^E \otimes \bar{\mathbf{a}}_3 = r_{ij}^{S,E} \mathbf{q}_i^{S,E} \otimes \bar{\mathbf{a}}_j \tag{3.24}$$

and the energy density can then be written in terms of the QR decomposition as was done in Jiang et al.[JGT17]

$$\chi(\mathbf{F}^{S,E}(\xi_1,\xi_2,0)) = \frac{\gamma}{2} \left(r_{13}^{S,E^2} + r_{23}^{S,E^2} \right) + f(r_{33}^{S,E}). \tag{3.25}$$

This follows because the normal and shear components of $\mathbf{a}_3^E$ can be written in terms of the basis vectors $\mathbf{q}_i^{S,E}$ from the QR decomposition $\mathbf{a}_3^E = r_{i3}^{S,E} \mathbf{q}_i^{S,E}$. With this convention, $s_3^E = r_{33}^{S,E}$ since $\text{span}\{\mathbf{a}_\alpha\} = \text{span}\{\mathbf{q}_\alpha^{S,E}\}$ and $\mathbf{q}_3^{S,E} = \mathbf{a}_3^{KL}$. Using $s_i^E = r_{i3}^{S,E}$ for conciseness

$$\boldsymbol{\tau}^S = \gamma s_i^E s_j^E \mathbf{q}_i^{S,E} \otimes \mathbf{q}_j^{S,E} + \left(f'(s_3^E) s_3^E - \gamma s_3^{E^2} \right) \mathbf{q}_3^{S,E} \otimes \mathbf{q}_3^{S,E}. \tag{3.26}$$

We provide a more detailed derivation of energies defined in terms of the QR decomposition and this specific case in Appendix §B.4.

3.3.4 Frictional contact yield condition and return mapping

With a continuum view of frictional contact, Coulomb friction defines a constraint on the types of stress that are admissible. This can be done concisely in terms of the Cauchy stress

$\boldsymbol{\sigma}$. This stress measure is defined through contact interactions internal to a continuum body [GS08]. Specifically, the contact force per unit area across a surface with normal $\mathbf{n}$ is $\boldsymbol{\sigma}\mathbf{n}$. In the shell, the contact direction is $\mathbf{a}_3^{KL}$. Coulomb friction places a constraint on the stress as

$$|\mathbf{t}_S| \leq -c_F \sigma_n \tag{3.27}$$

where $\boldsymbol{\sigma}\mathbf{a}_3^{KL} = \sigma_n \mathbf{a}_3^{KL} + \mathbf{t}_S$. Here $\boldsymbol{\sigma}\mathbf{a}_3^{KL}$ is contact force per unit area, $\sigma_n \mathbf{a}_3^{KL}$ is its normal component and $\mathbf{t}_S$ is the shearing component orthogonal to $\mathbf{a}_3^{KL}$. The condition in Equation (3.27) states that the magnitude of the shearing component can be no larger than a coefficient of friction times the normal component, with the convention that no shearing is allowed in the case of $\sigma_n > 0$ since this would be a separating rather than a compressive state. We note that each object can have its own coefficient of friction which provides a simple way of modeling interactions between many objects.

The Kirchhoff stress is related to the Cauchy stress as $\boldsymbol{\tau} = \det(\mathbf{F})\boldsymbol{\sigma}$. By design, the Kirchhoff-Love Kirchhoff stress has no component in the $\mathbf{a}_3^{KL}$ direction $\boldsymbol{\tau}^{KL}\mathbf{a}_3^{KL} = \mathbf{0}$. Therefore, the Coulomb friction constraint applies only to the shearing Kirchhoff stress $\boldsymbol{\tau}^S$. Using Equation (3.26) we can see that the continuum stress Coulomb friction condition is

$$\sqrt{{s_1^E}^2 + {s_2^E}^2} \leq \begin{cases} \frac{c_F k^c}{\gamma}\left(1 - s_3^E\right)^2, & 0 < s_3^E \leq 1 \\ 0, & s_3^E > 1 \end{cases} \tag{3.28}$$

Whereas the plastic constraint associated with denting involved the principle stresses of $\boldsymbol{\tau}^{KL}$, only the components s_i^E of the elastic $\mathbf{a}_3^E$ in the $\mathbf{q}_i^{S,E}$ basis are constrained under the Coulomb condition. It is satisfied with a return mapping of trial elastic $\mathbf{a}_3^{E^{\text{tr}}} = s_i^{E^{\text{tr}}}\mathbf{q}_i^{S,E}$ to the projected $\mathbf{a}_3^E = s_i^E \mathbf{q}_i^{S,E}$ where the trial and projected coefficients are related through

$$s_\alpha^E = \begin{cases} h(\mathbf{a}_3^{E^{\text{tr}}}) s_\alpha^{E^{\text{tr}}}, & 0 < s_3^{E^{\text{tr}}} \leq 1 \\ 0, & s_3^{E^{\text{tr}}} > 1 \end{cases}, \; s_3^E = \begin{cases} s_3^{E^{\text{tr}}}, & 0 < s_3^{E^{\text{tr}}} \leq 1 \\ 1, & s_3^{E^{\text{tr}}} > 1 \end{cases} \tag{3.29}$$

with

$$h(\mathbf{a}_3^{E^{\mathrm{tr}}}) = \begin{cases} \frac{c_F k^c \left(1 - s_3^{E^{\mathrm{tr}}}\right)^2}{\gamma \sqrt{s_1^{E^{\mathrm{tr}}2} + s_2^{E^{\mathrm{tr}}2}}}, & \sqrt{s_1^{E\mathrm{tr}2} + s_2^{E\mathrm{tr}2}} > \frac{c_F k^c}{\gamma} \left(1 - s_3^{E\mathrm{tr}}\right)^2 \\ 1, & \sqrt{s_1^{E\mathrm{tr}2} + s_2^{E\mathrm{tr}2}} \leq \frac{c_F k^c}{\gamma} \left(1 - s_3^{E\mathrm{tr}}\right)^2. \end{cases} \tag{3.30}$$

This is the projection from Jiang et al.[JGT17] where $0 < s_3^{E^{\mathrm{tr}}} \leq 1$ implies material is compressed from contact in the normal direction. In this case, the function h regulates the amount of shearing allowed relative to compression from the Coulomb constraint. In the case $s_3^{E^{\mathrm{tr}}} > 1$, material is separating in the normal direction and thus no resistance to shearing or compression is allowed.

3.4 Subdivision and B-spline FEM

The Kirchhoff-Love kinematics require higher regularity for midsurface interpolating functions in FEM calculations. This arises from the use of the normal $\mathbf{a}_3^{KL}$ in the definition of the kinematics in Equation (3.3) since the deformation gradient in the shell then depends on second order derivatives of the kinematics of the midsurface. Technically the requirement is H^2 regularity, meaning that the interpolating functions and all their derivatives of order less than or equal to two are square integrable over the midsurface. In practice this means that the interpolating functions must also have continuous first derivatives (C^1 continuous) over the midsurface. This is a challenging constraint on the interpolating functions. We represent the shell midsurfaces as Catmull-Clark subdivision surfaces since they posses the required regularity.

The Catmull-Clark subdivision scheme takes as input an arbitrary polygonal mesh and returns a subdivided, refined mesh. The input polygonal mesh is referred to as the control mesh, and the limiting result of the subdivision process yields a H^2 surface [CC78, Sta98]. As the output mesh from Catmull-Clark subdivisions only consists of quadrilateral faces, we may assume that all input meshes have quadrilateral faces by replacing the control mesh with its first subdivision if necessary.

We denote the world space locations of the control points by $\mathbf{x}_p$, where $p = 1, ..., n_p$ and n_p is the number of control points. We use $\mathbf{x}^{KL} = \big(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_p}\big)^T$ to denote the collection of all $\mathbf{x}_p$. The limiting surface from subdivision is represented as

$$\mathbf{x}(\mathbf{x}^{KL}, \xi_1, \xi_2) = \mathbf{x}_p N_p^{SD}(\xi_1, \xi_2),$$

where $N_p^{SD} \in H^2\Big(\omega \to [0,1]\Big)$ is the FEM basis weight function corresponding to the control point p. The N_p^{SD} have only local support and for each $(\xi_1, \xi_2) \in \omega$, only a sparse subset of $N_p^{SD}(\xi_1, \xi_2)$ are nonzero. We use the OpenSubdiv library to evaluate the basis functions $N_p^{SD}(\xi_1, \xi_2)$ and their first and second derivatives.

For each control mesh face, we sample rectangular quadrature points on either side of the face with $\xi_3 = -\frac{\tau}{4}$ and $\xi_3 = \frac{\tau}{4}$ for energy density evaluation. The generalized force on each of the control points is calculated as the negative derivative of the Kirchhoff-Love energy in Equation (3.10) which we approximate using quadrature

$$\Psi^{KL} = \sum_q V_q^0 \psi(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}^{KL})) \tag{3.31}$$

The derivatives satisfy

$$\mathbf{f}_p^{KL} = -\frac{\partial \Psi^{KL}(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}^{KL}))}{\partial \mathbf{x}_p} \tag{3.32}$$

$$= -\sum_q V_q^0 \frac{\partial \psi}{\partial \mathbf{F}}(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}^{KL}))) : \frac{\partial \mathbf{F}_q^{KL,E\mathrm{tr}}}{\partial \mathbf{x}_p}(\mathbf{x}^{KL}). \tag{3.33}$$

Here ξ_{q1}, ξ_{q2} are the locations of the quadrature points in parameter space and V_q^0 are their associated volumes. For each quadrature point q, the Kirchhoff-Love deformation gradient at mid-surface configuration $\mathbf{x}^{KL}$ is computed from

$$\mathbf{F}_q^{KL}(\mathbf{x}^{KL}) = \sum_{i=1}^{3} \mathbf{g}_{qi}(\mathbf{x}^{KL}) \otimes \bar{\mathbf{g}}_q^i. \tag{3.34}$$

Furthermore, in Equation (3.33),

$$\frac{\partial \psi}{\partial \mathbf{F}}(\mathbf{F}_q^{KL,E\text{tr}}(\mathbf{x}^{KL})) = \boldsymbol{\tau}^{KL}(\mathbf{F}_q^{KL,E\text{tr}}(\mathbf{x}^{KL}))\left(\mathbf{F}_q^{KL,E\text{tr}}(\mathbf{x}^{KL})\right)^T$$

where $\boldsymbol{\tau}^{KL}$ is from Equation (3.12). This relation follows from the definition of the first Piola-Kirchhoff stress and its relation to the Kirchhoff stress [BW08].

The trial elastic deformation $\mathbf{F}^{KL,E\text{tr}}$ and its derivative with respect to control points $\frac{\partial \mathbf{F}_q^{KL,E\text{tr}}}{\partial \mathbf{x}_p}(\mathbf{x}^{KL})$ are computed assuming no further plastic flow over the time step

$$\mathbf{F}_q^{KL,Etr} = \mathbf{F}_q^{KL}\mathbf{F}_q^{KL,P,n-1} \tag{3.35}$$

$$\frac{\partial \mathbf{F}_q^{KL,E\text{tr}}}{\partial \mathbf{x}_p}(\mathbf{x}^{KL}) = \frac{\partial \mathbf{F}_q^{KL}}{\partial \mathbf{x}_p}(\mathbf{x}^{KL})\mathbf{F}_q^{KL,P,n-1} \tag{3.36}$$

Note that when calculating the generalized force in Equation (3.32)-(3.33), $\mathbf{F}^{KL,E\text{tr}}$ is used even though the associated stress may not satisfy the yield criteria. This is a consequence of the variational FEM discretization of the analogous formula for the stress in terms of derivative of the strain energy density[BW08]. We provide the calculation of $\mathbf{F}_q^{KL}(\mathbf{x}^{KL})$ and $\frac{\partial \mathbf{F}_q^{KL}}{\partial \mathbf{x}_p}(\mathbf{x}^{KL})$ in Appendix §B.2.

3.5 MPM Discretization

We use MPM to discretize our elastoplastic model for frictional contact. We represent the shell using particles connected with subd interpolation as in §3.4. That is, we consider the subd FEM control point as particles in a MPM method. This allows us to resolve contact and collision automatically through the elastoplastic constitutive behavior when we transfer to the background grid. There is no need for any collision detection or resolution other than that inherent in the MPM discretization of the continuum model. Furthermore, our approach naturally allows for coupling with materials (e.g. granular sand, snow and soil) simulated with MPM.

MPM is a hybrid Lagrangian/Eulerian approach. However, the primary representation of material for MPM is the Lagrangian state. At time t^n, we store particle position $\mathbf{x}_p^n$, velocity $\mathbf{v}_p^n$, initial mass m_p, initial volume V_p^0, affine velocity $\mathbf{C}_p^n$ for all materials in the simulation. Similar to Jiang et al.[JGT17], we classify particles as either: (i) traditional MPM particles, (ii) subd particles or (iii) continuum shell shearing/compression particles. Particles of type (i) are used for coupling with traditional MPM materials like sand or snow. Types (ii) and (iii) are associated with elasticity and frictional contact respectively in the subd shell mesh. Furthermore, particles of type (ii) are control vertices in $\mathbf{x}^{KL}$ (see §3.4) for the subd shell and particles of type (iii) are quadrature points for the shearing component of the energy in Equation (3.10) and lie on the subd surface. For particles of type (i), we store the elastic deformation gradient $\mathbf{F}_p^{E,n}$. For particles of type (iii), we store the time t^n elastic shearing $\mathbf{a}_{p3}^E$ and the parameters in the mid-surface (ξ_{p1}, ξ_{p2}) associated with the particle. As in Jiang et al.[JGT17], we use the notation $\mathcal{I}^{(i)}$,$\mathcal{I}^{(ii)}$,$\mathcal{I}^{(iii)}$ to represent the sets of particle indices of types (i), (ii) and (iii) respectively. At each of the quadrature points used in the Kirchhoff-Love energy, we store the deformation gradient and its elastic and plastic components $\mathbf{F}_q^{KL,n}$, $\mathbf{F}_q^{KL,E,n}$, $\mathbf{F}_q^{KL,P,n}$, the reference contravariant basis vectors $\bar{\mathbf{g}}_q^i$ needed for deformation gradient computation, and the mid-surface parameters (ξ_{p1}, ξ_{p2}) associated with the point. Although these quadrature points are not MPM particles and are not used in transfers to and from the grid etc., we additionally use $\mathcal{I}^{(iv)}$ to denote the collection of quadrature points used in the Kirchhoff-Love energy. We illustrate all particle and quadrature point types in Figure 3.6.

In MPM, the Eulerian grid can be viewed as an auxiliary structure for updating the Lagrangian state. We first transfer the particle mass and momentum state to an equivalent grid counterpart. We use $m_{\mathbf{i}}^n$ to denote the mass of Eulerian grid node $\mathbf{x}_{\mathbf{i}}$ at time t^n, $\mathbf{v}_{\mathbf{i}}^n$ to denote its velocity and $\mathbf{p}_{\mathbf{i}}^{n+1}$ to denote its linear momentum after the grid update. The grid momentum is updated from the force defined as the gradient of the potential energy with respect to grid node motion. The motion of the grid is then interpolated to the particles to update the Lagrangian state without ever actually moving grid nodes. Our approach is ultimately very similar to other MPM methods that define forces from a notion of potential energy [YSB15, DB16, SSC13, JSS15, KGP16] and particularly Jiang et al.[JGT17]. We briefly discuss aspects common to the approach of Jiang et al.[JGT17] and discuss our novel modifications needed for subd shells in more detail.

3.5.1 Grid transfers: particle to grid

To update the Lagrangian state, we transfer mass and momentum from particles $\mathbf{x}_p^n$ to the grid nodes $\mathbf{x_i}$ using APIC [JSS15].

$$m_{\mathbf{i}}^n = \sum_p w_{\mathbf{i}p}^n m_p \tag{3.37}$$

$$\mathbf{v}_{\mathbf{i}}^n = \frac{1}{m_{\mathbf{i}}^n} \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{C}_p^n(\mathbf{x}_{\mathbf{i}}^n - \mathbf{x}_p^n)) \tag{3.38}$$

Here $w_{\mathbf{i}p}^n = N(\mathbf{x}_p^n - \mathbf{x_i})$ is the weight of interaction between particle $\mathbf{x}_p^n$ and grid node $\mathbf{x_i}$. The $N(\mathbf{x})$ are linear, quadratic or cubic B-spline kernels used for interpolation over the grid. $\mathbf{v}_p^n$ and $\mathbf{C}_p^n$ define an affine notion of velocity local to the particle.

3.5.2 Grid momentum update

The grid momentum update uses the updated Lagrangian view of the governing physics [BLM13, FGG17b]. The grid at time t^n, after transferring state from the Lagrangian particles, is an alternative Lagrangian mesh with degrees of freedom $\mathbf{x_i}$, $\mathbf{v}_{\mathbf{i}}^n$ and mass $m_{\mathbf{i}}^n$. Its update is derived from the Lagrangian FEM discretization of a problem with a notion of potential energy. The internal force is the negative gradient of the potential energy with respect to positional changes. Using $\mathbf{x}_{\mathbf{i}}^{n+1}$ and $\mathbf{p}_{\mathbf{i}}^{n+1}$ to denote the new position and linear momentum state after the time step, the grid discretization has the form

$$\mathbf{x}_{\mathbf{i}}^{n+1} = \mathbf{x_i} + \frac{\Delta t}{m_{\mathbf{i}}^n} \mathbf{p}_{\mathbf{i}}^{n+1} \tag{3.39}$$

$$\mathbf{p}_{\mathbf{i}}^{n+1} = m_{\mathbf{i}}^n \mathbf{v}_{\mathbf{i}}^n - \Delta t \frac{\partial \Psi}{\partial \mathbf{x_i}}(\mathbf{x}^*) + \Delta t m_{\mathbf{i}}^n \mathbf{g} \tag{3.40}$$

where $\Psi(\mathbf{x})$ is the potential energy which depends on the positional state where we use $\mathbf{x}^* = \left(\mathbf{x}_{\mathbf{i}^1}^*, \mathbf{x}_{\mathbf{i}^2}^*, \ldots\right)^T$ to denote the vector of all grid node positions. In the case of symplectic

Euler integration, $\mathbf{x}_{\mathbf{i}}^* = \mathbf{x}_{\mathbf{i}}$ and in the case of backward Euler, $\mathbf{x}_{\mathbf{i}}^* = \mathbf{x}_{\mathbf{i}}^{n+1}$. We note that the grid nodes are not actually moved from $\mathbf{x}_{\mathbf{i}}$ to $\mathbf{x}_{\mathbf{i}}^{n+1}$. Instead, the motion of the grid is interpolated to the particles (see §3.5.3).

The potential energy Ψ is a sum of the contributions from the shell Ψ^{CS} and from traditional MPM particles ψ^M used for coupling multiple materials.

$$\Psi(\mathbf{x}^*) = \sum_{p\in\mathcal{I}^{(i)}} \psi^M\left(\mathbf{F}_p^{E,\mathrm{tr}}(\mathbf{x}^*)\right)V_p^0 + \Psi^{CS}(\mathbf{x}^*) \tag{3.41}$$

$$\begin{aligned}\Psi^{CS}(\mathbf{x}^*) = \sum_{p\in\mathcal{I}^{(iii)}} \chi\Big(&\mathbf{a}_{p\alpha}(\mathbf{x}^{KL}(\mathbf{x}^*))\otimes\bar{\mathbf{a}}_{p\alpha}\\ &+\mathbf{a}_{p3}^{E,\mathrm{tr}}(\mathbf{x}^{KL}(\mathbf{x}^*))\otimes\bar{\mathbf{a}}_{p3}\Big)V_p^0\\ +\sum_{q\in\mathcal{I}^{(iv)}}\psi&\left(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}^{KL}(\mathbf{x}^*))\right)V_q^0.\end{aligned} \tag{3.42}$$

Here ψ^M is the contribution from the standard MPM potential discretization (see e.g. Stomakhin et al.[SSC13]) and Ψ^{CS} is the contribution from the continuum shell. An advantage of the MPM approach is that coupling is achieved between any materials whose constitutive behaviors can be defined from potential energies. With any such models, coupling is achieved by first representing the motion of the materials in a Lagrangian way (e.g. discrete particles or Lagrangian meshes) and defining their motion and the way it effects their potential energy in terms of interpolation from the grid. With this model, coupling is a simple as defining the total potential energy as the sum of the varied materials.

The energy Ψ^{CS} is the sum of the discretization of the Kirchhoff-Love component in Equation (3.10) given in Equation (3.31) and the frictional contact energy in Equation (3.23) obtained from the quadrature points $q \in \mathcal{I}^{(iv)}$ and $p \in \mathcal{I}^{(iii)}$ respectively. We highlight the dependence of these potentials on the grid motion $\mathbf{x}^*$. For particles of type (i), this dependence follows from the updated Lagrangian formulation

$$\mathbf{F}_p^{E,\mathrm{tr}}(\mathbf{x}^*) = \left(\sum_{\mathbf{i}} \mathbf{x}_{\mathbf{i}}^* \otimes \nabla w_{\mathbf{i}p}^n\right)\mathbf{F}_p^{E,n} \tag{3.43}$$

Here $\nabla w^n_{\mathbf{i}p} = \nabla N(\mathbf{x}^n_p - \mathbf{x}_\mathbf{i})$ is the gradient of the grid interpolating function (or weight gradient) and $\left(\sum_\mathbf{i} \mathbf{x}^*_\mathbf{i} \otimes \nabla w^n_{\mathbf{i}p}\right)$ represents deformation induced by the grid motion $\mathbf{x}^*$. For particles of type (iii), the dependence follows from the updated Lagrangian

$$\mathbf{a}^{E,\mathrm{tr}}_{p3}(\mathbf{x}^*) = \left(\sum_\mathbf{i} \mathbf{x}^*_\mathbf{i} \otimes \nabla w^n_{\mathbf{i}p}\right) \mathbf{a}^{E,n}_{p3} \tag{3.44}$$

and from interpolation the $\mathbf{x}^{KL}(\mathbf{x}^*)$ in Equation (3.45) in $\mathbf{a}_{p\alpha}(\mathbf{x}^{KL}(\mathbf{x}^*))$ (Appendix §B.6). Following the approaches in Jiang et al.[JSS15, JGT17], the mid-surface control points for the shell are interpolated from the grid degrees of freedom as

$$\mathbf{x}^*_p = \sum_\mathbf{i} \mathbf{x}^*_\mathbf{i} w^n_{\mathbf{i}p}, \quad p \in \mathcal{I}^{(ii)}. \tag{3.45}$$

This interpolation also affects the discrete Kirchhoff-Love term through quadrature points $q \in \mathcal{I}^{(iv)}$.

Taking the $\mathbf{x}^*$ dependence into account and using the chain rule, the potential energy based forces obtained from the gradient of Ψ with respect to $\mathbf{x}^*$ are

$$\frac{\partial \Psi}{\partial \mathbf{x}_\mathbf{i}}(\mathbf{x}^*) = \mathbf{f}^{(i)}_\mathbf{i}(\mathbf{x}^*) + \mathbf{f}^{(ii)}_\mathbf{i}(\mathbf{x}^*) + \mathbf{f}^{(iii)}_\mathbf{i}(\mathbf{x}^*) \tag{3.46}$$

$$\mathbf{f}^{(i)}_\mathbf{i}(\mathbf{x}^*) = \sum_{p \in \mathcal{I}^{(i)}} \frac{\partial \psi^M}{\partial \mathbf{F}^E}(\mathbf{F}^{E,\mathrm{tr}}_p(\mathbf{x}^*)) \mathbf{F}^{E,n^T}_p \nabla w^n_{\mathbf{i}p} V^0_p \tag{3.47}$$

$$\mathbf{f}^{(ii)}_\mathbf{i}(\mathbf{x}^*) = \sum_{p \in \mathcal{I}^{(ii)}} w^n_{\mathbf{i}p} \mathbf{f}^{KL}_p(\mathbf{x}^{KL}(\mathbf{x}^*)) \tag{3.48}$$

$$\mathbf{f}^{(iii)}_\mathbf{i}(\mathbf{x}^*) = \sum_{p \in \mathcal{I}^{(iii)}} \boldsymbol{\tau}^S_p \tilde{\mathbf{a}}^\beta_p : \frac{\partial \mathbf{a}_{p\beta}}{\partial \mathbf{x}_p} w^n_{\mathbf{i}p} + \boldsymbol{\tau}^S_p \tilde{\mathbf{a}}^3_p : \nabla w^n_{\mathbf{i}p} \mathbf{a}^{E,n}_{p3} \tag{3.49}$$

In Equation (3.48), $\mathbf{f}^{KL}_p$ is the generalized Kirchhoff-Love force from Equation (3.32). In Equation (3.49), the stress $\boldsymbol{\tau}^S_p$ is from Equation (3.26) and the vector $\tilde{\mathbf{a}}^3_p$ is the third contravariant basis vector with respect to the covariant basis $\{\mathbf{a}_\alpha(\mathbf{x}^*), \mathbf{a}^{E,\mathrm{tr}}_3(\mathbf{x}^*)\}$. We refer to Appendix §B.1 for this derivation.

3.5.3 Grid transfers: grid to particle

The grid to particle transfer defines the time t^{n+1} affine velocity local to particle $\mathbf{x}_p^n$ in terms of $\mathbf{v}_p^{n+1}$ and $\mathbf{C}_p^{n+1}$ from

$$\mathbf{v}_p^{n+1} = \sum_{\mathbf{i}} w_{\mathbf{i}p}^n \frac{\mathbf{p}_{\mathbf{i}}^{n+1}}{m_{\mathbf{i}}^n} \tag{3.50}$$

$$\tilde{\mathbf{C}}_p^{n+1} = \frac{12}{\Delta x^2(d+1)} \sum_{\mathbf{i}} w_{\mathbf{i}p}^n \frac{\mathbf{p}_{\mathbf{i}}^{n+1}}{m_{\mathbf{i}}^n} \otimes (\mathbf{x}_{\mathbf{i}}^n - \mathbf{x}_p^n) \tag{3.51}$$

$$\mathbf{C}_p^{n+1} = (1-\nu)\,\tilde{\mathbf{C}}_p^{n+1} + \frac{\nu}{2}\left(\tilde{\mathbf{C}}_p^{n+1} - \tilde{\mathbf{C}}_p^{n+1T}\right) \tag{3.52}$$

Here d is the B-spline degree ($d=3$ for cubic b-spline interpolation, $d=2$ for quadratic B-spline interpolation) and Δx is the Eulerian grid spacing. ν is the explicit damping coefficient from Jiang et al.[JGT17] where $\nu = 0$ is completely undamped and $\frac{1}{2}\left(\tilde{\mathbf{C}}_p^{n+1} - \tilde{\mathbf{C}}_p^{n+1T}\right)$ is the RPIC transfer from Jiang et al.[JSS15].

3.5.4 Update positions and trial elastic state

For particles of type (i) and (ii), positions are moved with the interpolated grid node velocities. For particles of type (iii), positions are updated based on interpolation from updated particles of type (ii).

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1} = \sum_{\mathbf{i}} \mathbf{x}_{\mathbf{i}}^{n+1} w_{\mathbf{i}p}^n,\ p \in \mathcal{I}^{(i)} \cup \mathcal{I}^{(ii)} \tag{3.53}$$

$$\mathbf{x}_p^{n+1} = \sum_{p^{(ii)} \in \mathcal{I}^{(ii)}} \mathbf{x}_{p^{(ii)}}^{n+1} N_{p^{(ii)}}^{SD}(\xi_{p1}, \xi_{p2}),\ p \in \mathcal{I}^{(iii)}. \tag{3.54}$$

We first assume there was no additional plastic flow over the time step and consider a trial state of elastic deformation. For particles of type (i) and (iii), the trial elastic deformation $\mathbf{F}_p^{E,\mathrm{tr}}$ and $\mathbf{a}_{p3}^{E,\mathrm{tr}}$ are computed as in Equations (3.43) and (3.44) respectively with $\mathbf{x}_{\mathbf{i}}^* = \mathbf{x}_{\mathbf{i}}^{n+1}$. For Kirchhoff-Love quadrature points $q \in \mathcal{I}^{(iv)}$, the trial elastic deformation gradient $\mathbf{F}_q^{KL,E\mathrm{tr}}$ is computed from Equation (3.35) where $\mathbf{x}^{KL}(\mathbf{x}^*)$ is interpolated as in Equation (3.45) with $\mathbf{x}_{\mathbf{i}}^* = \mathbf{x}_{\mathbf{i}}^{n+1}$.

3.5.5 Update plasticity

The assumption of no plastic flow over the time step is often safe. However, if the trial state of elastic stresses are not inside the yield surfaces associated with denting, frictional contact, etc. then they must be projected to satisfy the constraint. For particles $p \in \mathcal{I}^{(i)}$, $\mathbf{F}_p^{E,\mathrm{tr}}$ is projected to $\mathbf{F}_p^{E,n+1}$ in accordance with whichever yield surface is being used (e.g. the Drucker-Prager law in Klár et al.[KGP16]). For quadrature points $q \in \mathcal{I}^{(iv)}$, $\mathbf{F}_q^{E,\mathrm{tr}}$ and $\mathbf{F}_q^{P,\mathrm{tr}}$ are projected to $\mathbf{F}_q^{E,n+1}$ and $\mathbf{F}_q^{P,n+1}$ in accordance with the denting return mapping in §3.3.2. Lastly, the $\mathbf{a}_{p3}^{E,\mathrm{tr}}$ are projected to $\mathbf{a}_{p3}^{n+1}$ in accordance with the frictional contact return mapping in Equation (4.23).

3.6 Results

We demonstrate the efficacy of our method on a number of representative examples that exhibit appreciable bending and persistent self-collision and show that our method automatically allows for coupling with granular materials. Furthermore, we demonstrate the range of behaviors that are possible with the parameters in our model. We list the runtime performance for all of our examples in Table 3.1. All simulations were run on an Intel Xeon E5-2687W v4 system with 48 hyperthreads and 128GB of RAM. We report the timing in terms of average seconds of computation per frame. We chose Δt in an adaptive manner that is restricted by a CFL condition when the particle velocities are high. In all of our simulations we use a CFL number equal to 0.3, i.e., we do not allow particles to move further than $0.3\Delta x$ in a time step.

3.6.1 Effect of shell thickness

We control the bending stiffness of the shell by varying the thickness τ. In Figure 1.1, six cylinders with increasing thickness from left to right free-fall and drop on the ground. In Figure 3.8, four cylinders of decreasing thickness from left to right buckle under lateral pressure and exhibit characteristic buckling patterns. In Figure 3.7, ribbons of varying thickness are planted in plates and twisted to produce interesting buckling phenomena.

3.6.2 Woven fabrics

We demonstrate that our method supports any potential function in the Kirchhoff-Love shell model. In particular, we implement the data-driven orthotropic model for woven fabrics from Clyde et al.[CTT17] with parameters fitted from experimental data. In Figure. 3.9a and 3.9b, we twist and compress sleeves made of denim and silk. In Figure. 3.10a and 3.10b, we suspend squares of silk and denim which then collide with moving spheres. Our model accurately captures the behaviors of these real world materials.

3.6.3 Self collisions

Our model successfully resolves self-collision without any use of collision detection or constraint modeling outside the MPM discretization. We demonstrate this in a number of representative scenarios. In Figure 3.2, the spheres and the diving boards, both modeled as shells, collide with each other. In Figure 1.1 and Figure 3.12, we demonstrate self-collisions resolution for clothing simulation stress tests. In Figure 3.5, four decks of cards collide and then slide against each other to demonstrate the effect of varying friction coefficients.

3.6.4 Plasticity for denting

Our method naturally incorporates the effect of plasticity in the shell. In Figure 3.4, three cylinders with different yield stress are twisted and then released. By changing the yield stress, we are able to control the amount of denting. In Figure 3.13, a square sheet of metal is compressed and then dented with a rod. The effect of plasticity creates permanent buckling and denting deformation.

3.6.5 Two-way coupling

Our MPM approach automatically resolves coupling of different materials. In Figure 1.1, a cup is filled with slush and then released and toppled. The cup is modeled as a shell and the slush is modeled as in Stomakhin et al.[SSC13]. This example demonstrates that our method successfully resolves the interactions between two different materials of millions of particles with moderate computation cost.

3.6.6 Resolution refinement

In Figure 3.11 we examine the behavior of our method under refinement of grid and subd mesh spatial resolution. This refinement study is done on a sleeve-buckling simulation with boundary conditions compressing the material at top and bottom. As the spatial resolution is increased, the simulation converges to the characteristic buckling pattern that is expected.

3.6.7 Bending with Jiang et al.

We demonstrate the failure of the Jiang et al. [JGT17] model in achieving significant bending resistance. In Figure 3.15 we compare our model with the Jiang et al. generalized to bending with the addition of bending springs. The frictional contact model in Jiang et al. [JGT17] was not designed for bending resistance, however it is possible to simply add bending cross springs to their model even though it violates the stress assumptions. We show that this is not capable of generating significant resistance to bending whereas our approach is designed to support stiff shells and thin membranes.

3.7 Discussion and Limitations

While our method can efficiently simulate thin shells with extreme contact and collision, there are a number of notable limitations. First, we have the same artifacts as Jiang et al. [JGT17], namely visible separation if Δx is too large, persistent wrinkles if subd mesh resolution is too high relative to the grid resolution and self-penetration if the resolution is

	Figure	seconds/frame	Element #	Particle #	Δx
Cup of Slush	1.1	273	19.5K	3.1M	0.04
Shirt Twister	1.1	188	168K	504K	0.005
Six Cylinders	1.1	2/2/2/2/2/4	20K	60K	0.025
Walk Cycle	1.1	75	33K	100K	10
Silk Curtain	3.10a	167	75K	227K	0.004
Denim Curtain	3.10b	3	8K	25K	0.012
Pants Twister	3.12	78	131K	393K	0.005
Silk Twister	3.9a	47	63K	315K	0.02
Denim Twister	3.9b	3	15.8K	47K	0.04
Spheres On Diving Board	3.2	87	150K	450K	0.027
Playing Cards	3.5	55	23K	115K	0.02
Plastic Twister	3.4	< 1	5K	14K	0.06
Sleeves (Yellow)	3.8	97	126K	378K	0.01
Sleeves (Others)	3.8	8	31K	93K	0.02
Fixed Ribbons	3.7	3/8/30	12K	93K	0.02
Free Ribbons	3.7	4/4/7	12K	85K	0.02
Denting with Rod	3.13	< 1	5K	15K	0.01

Table 3.1: All simulations were run on Intel Xeon E5-2687W v4 system with 48 hyperthreads and 128GB of RAM. Element # denotes number of quadrilaterals. Particle # denotes the number of type (*i*), (*ii*) and (*iii*) particles.

too low relative to the grid. Also, the time step size is generally smaller than those used for membranes in Jiang et al. [JGT17]. This is due to the added stiffness associated with shell thickness and bending. With MPM, the increase in time step size with implicit time stepping is bounded above since particles cannot move more than a grid cell in a time step without causing bunching, self collision or material inversion. Therefore the demand on the efficiency of nonlinear solver for the implicit systems is very high. Unfortunately this demand is difficult to meet since the nonlinear systems have non-symmetric linearizations that result from the plasticity [KGP16]. “Lagging” the plasticity as in Stomakhin et al.[SSC13] provides a symmetric linearization but can cause cohesion artifacts that are unacceptable for frictional contact applications. Development of a solver that is more efficient than Newton’s method with GMRES for the linearized systems is an interesting area of future work.

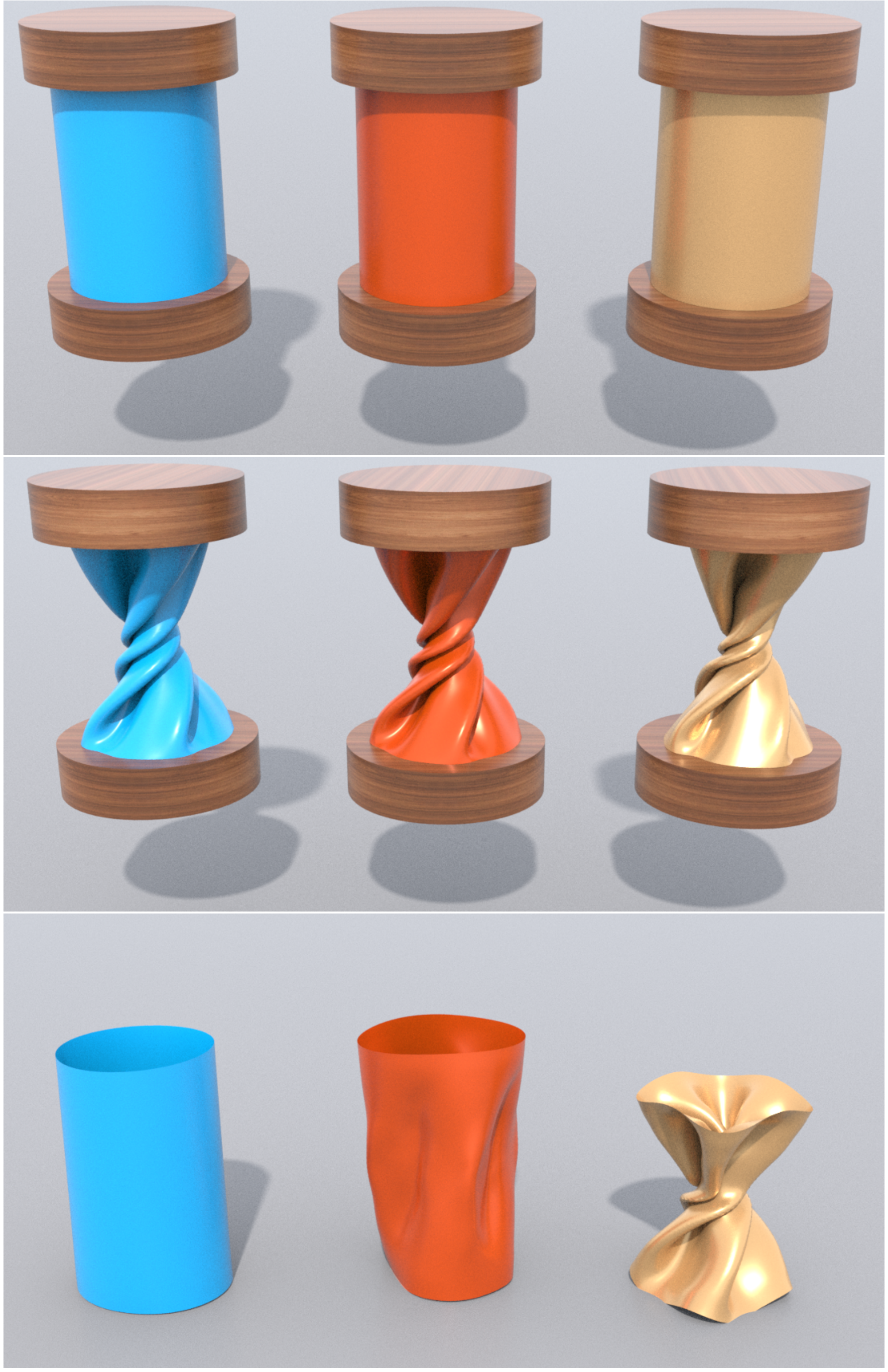

Figure 3.4: **Plastic shell deformation**. The effect of the yield condition in Equation (3.13) is shown here with decreasing values of the coefficient c_{vM} (from left to right). Larger values correspond to a larger stress needed for before denting plasticity is induced. The cylinders are twisted and then dropped to the ground to illustrate the plastic deformation.

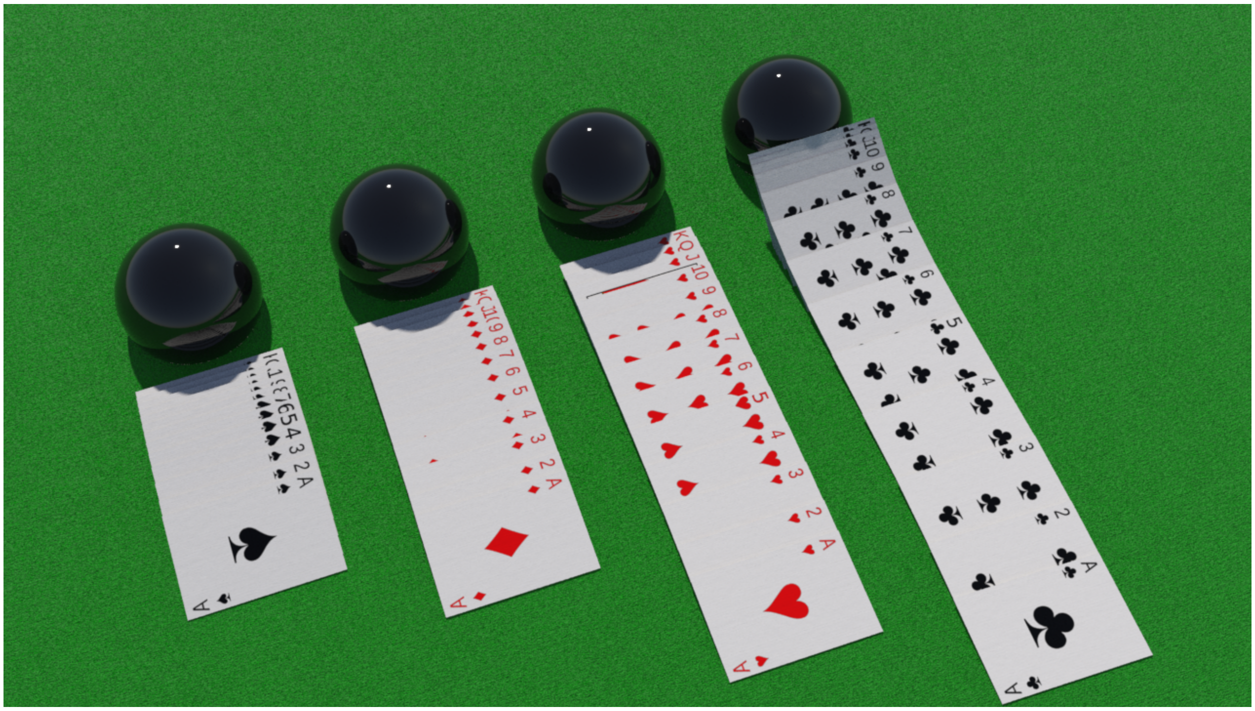

Figure 3.5: **Variation in Coulomb friction coefficient**. The effect of the friction parameter c_F can be seen in this card comparison. By decreasing c_F (from left to right) we demonstrate a range of surface frictions.

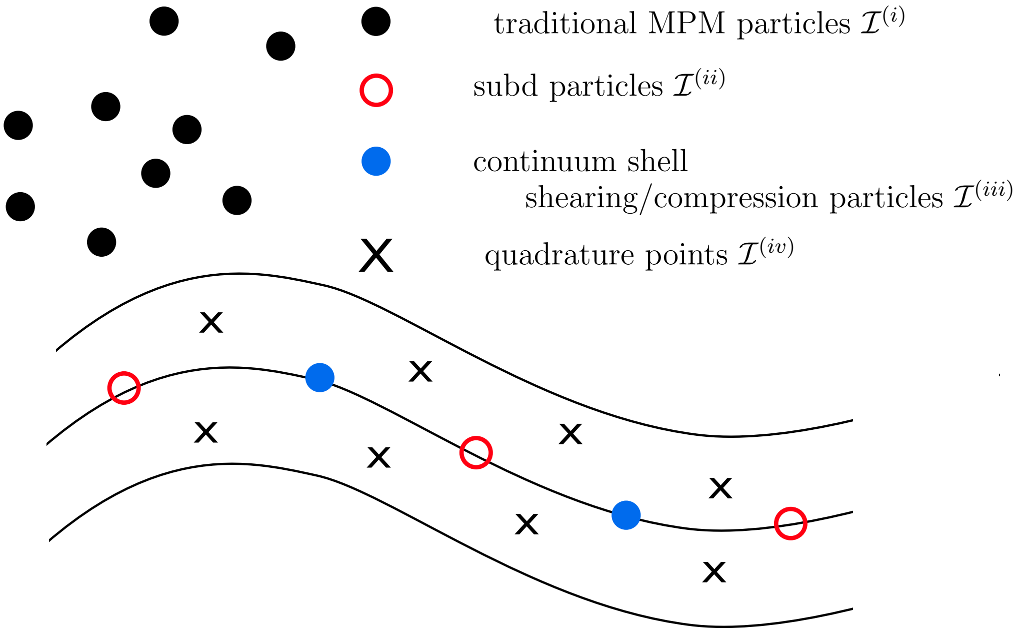
traditional MPM particles $\mathcal{I}^{(i)}$
subd particles $\mathcal{I}^{(ii)}$
continuum shell shearing/compression particles $\mathcal{I}^{(iii)}$
quadrature points $\mathcal{I}^{(iv)}$

Figure 3.6: **Particle type classification**. A schematic illustration of the different types of MPM particles and quadrature points.

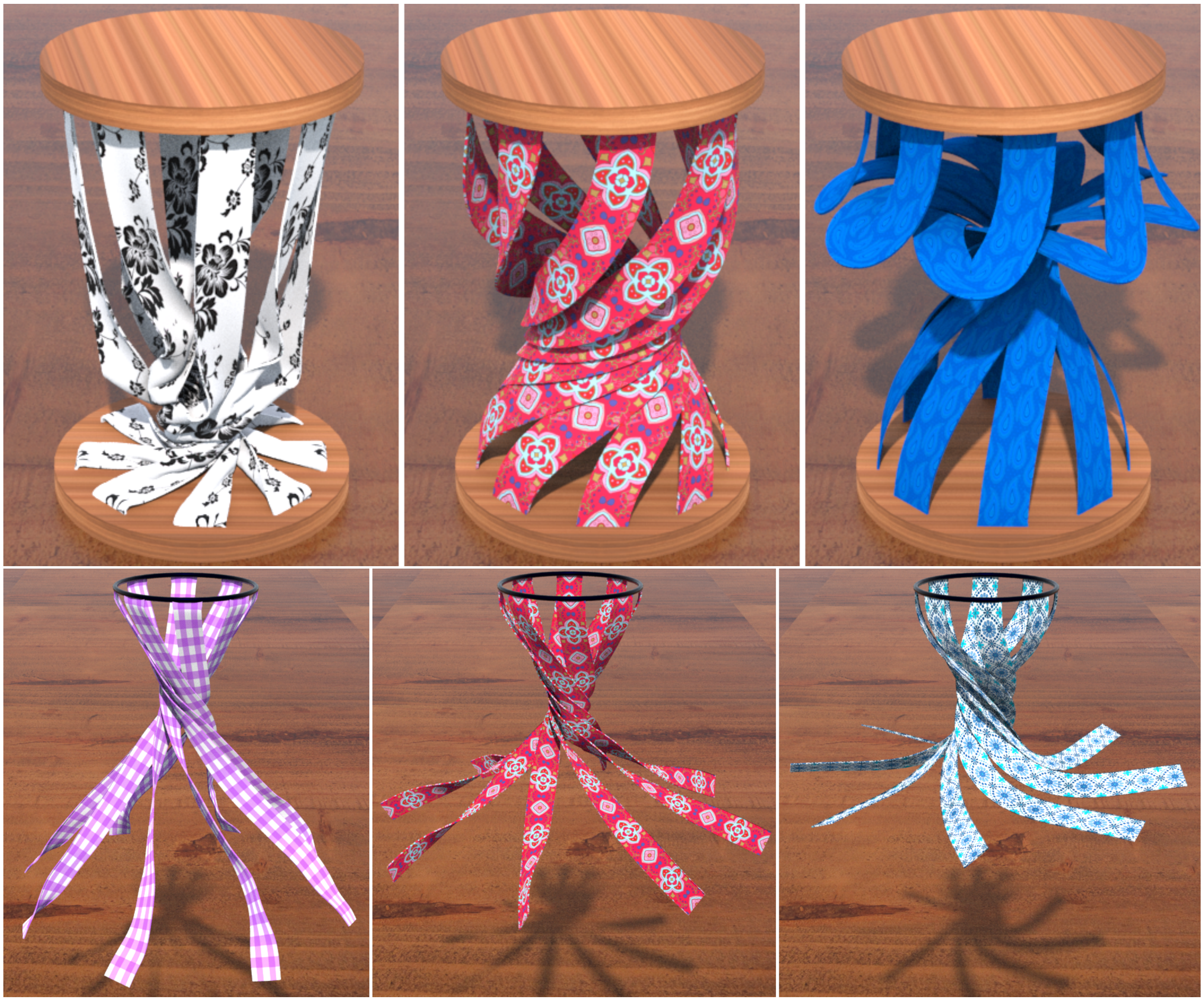

Figure 3.7: **Ribbons**. We illustrate interesting dynamics achieved from colliding ribbons with increasing thickness (from left to right).

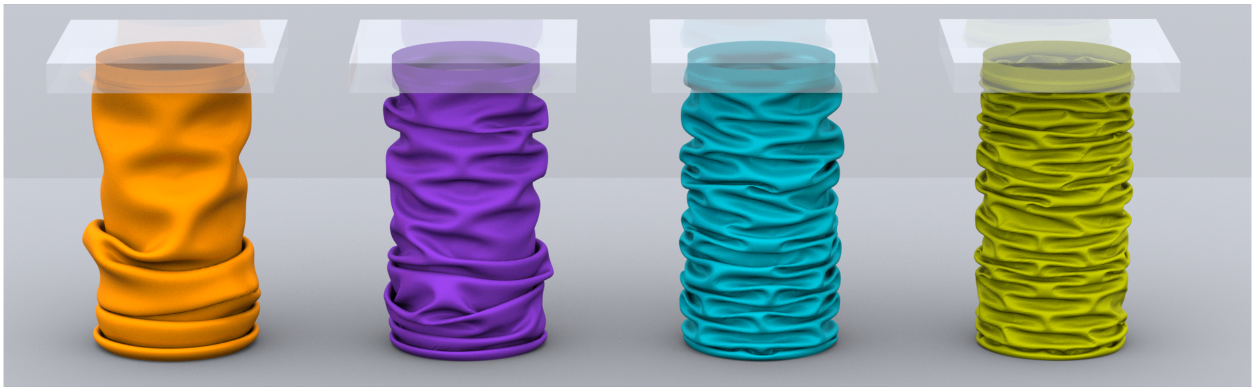

Figure 3.8: **Variation in shell thickness**. We demonstrate the effect of the shell thickness parameter in a compression comparison.

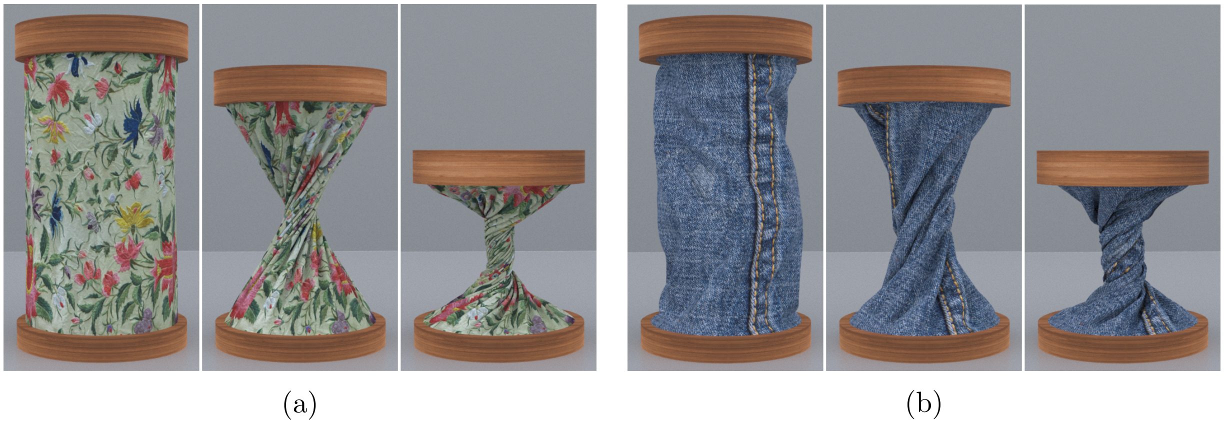

Figure 3.9: **Twisting Orthotropic Model**. Using the data-driven model of Clyde et al. [CTT17] for woven materials, the characteristic wrinkling of silk (left) and denim (right) is obtained. Our method naturally resolves the many self-collisions induced by the twisting boundary conditions.

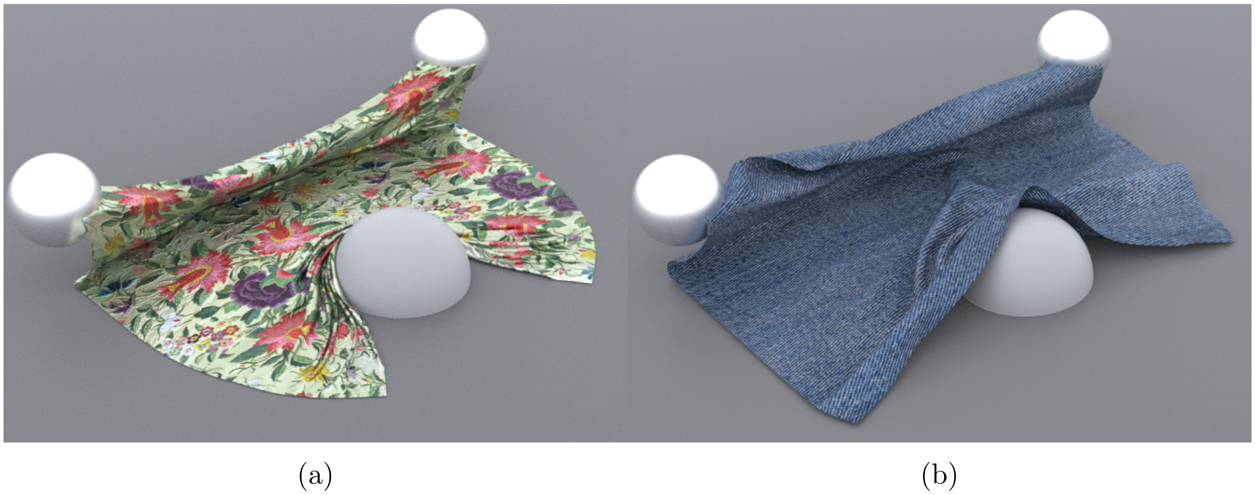

Figure 3.10: **Orthotropic Model**. A range of materials can be simulated with our continuum shell formulation. Here we use the data-driven model of Clyde et al. [CTT17] for woven silk (left) and denim (right) materials. The model naturally allows for characteristic buckling and wrinkling behaviors in this object collision test.

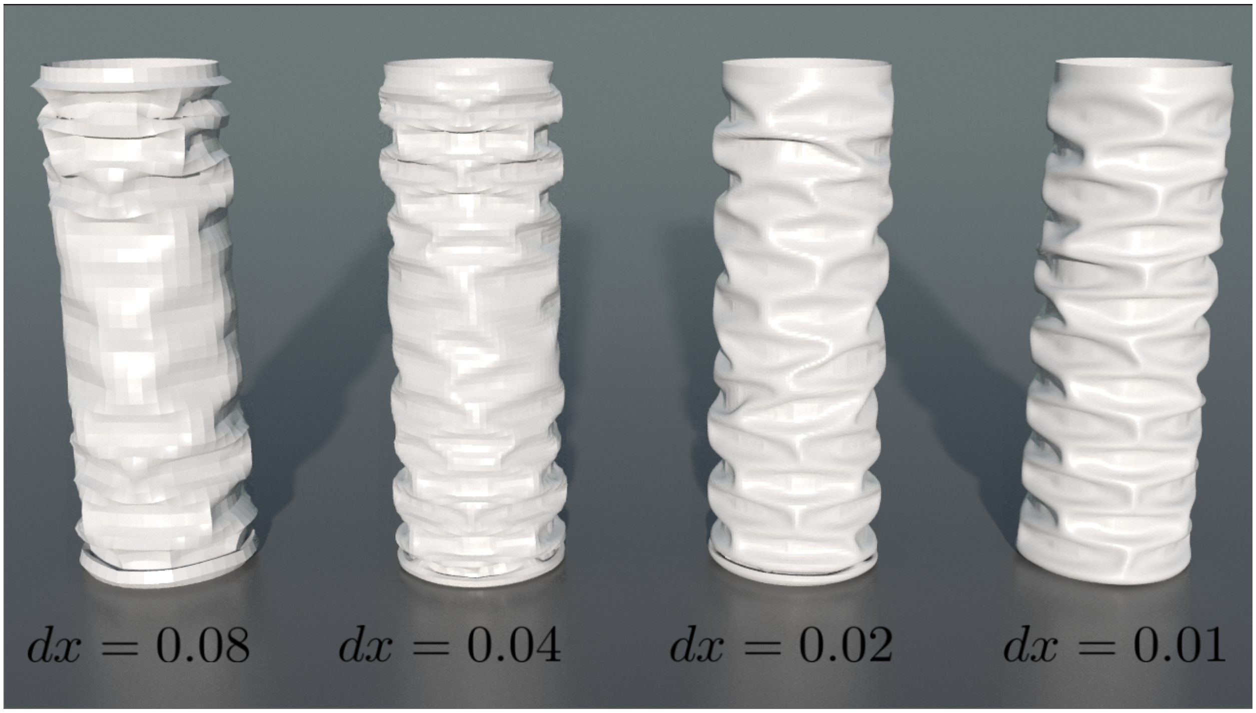

Figure 3.11: **Convergence under spatial refinement**. We demonstrate that our method converges under refinement of grid and subd mesh spatial resolution in this buckling example. The simulations have increasing spatial resolution from left to right.

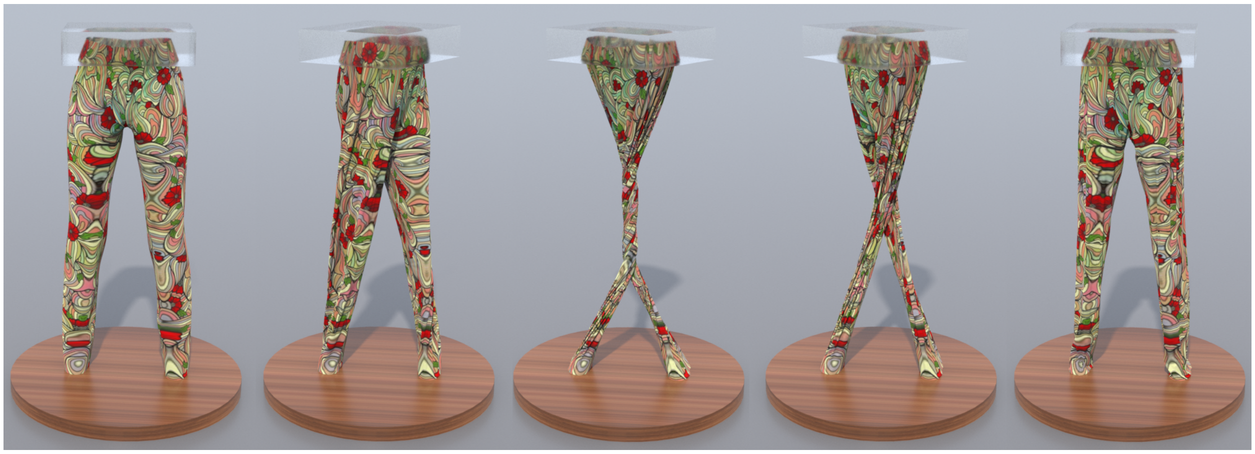

Figure 3.12: **Pants twister**. Our approach works for clothing simulation with many self collisions as shown here in the legs of a twisted pair of pants. The subdivision mesh for the pants has 393K control points and the simulation runs at 78s per frame.

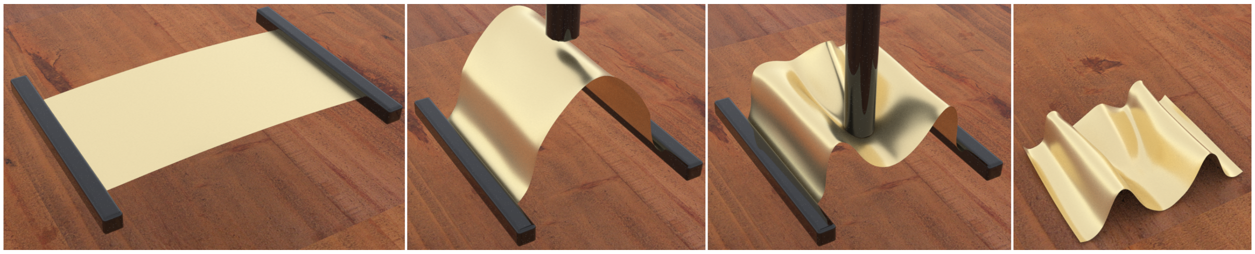

Figure 3.13: **Denting**. We demonstrate plastic deformation of foil induced by object collision.

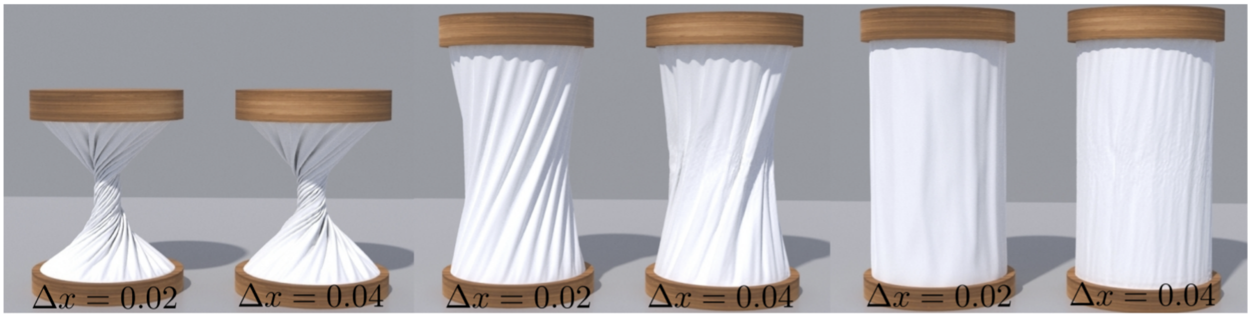

Figure 3.14: **Grid resolution dependent wrinkling.** Our method suffers from persistent wrinkling if the subd mesh resolution is too high relative to the grid resolution. We demonstrate this phenomenon here with a cloth twisting comparison example. In both examples, the subd mesh $\Delta x = 0.02$. The example on the left has grid $\Delta x = 0.02$ whereas the one on the right has grid $\Delta x = 0.04$.

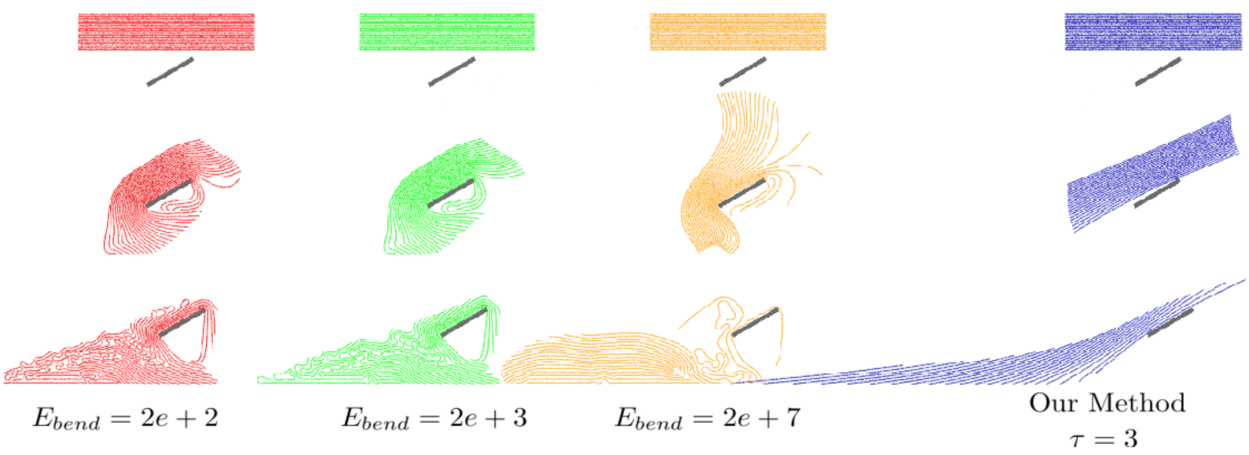

Figure 3.15: **Jiang et al. [JGT17] comparison**. We demonstrate that only moderate bending is possible with the approach of Jiang et al. [JGT17]. Our approach allows for a much wider range of bending resistance.

CHAPTER 4

Simulation for Volumetric Objects with Frictional Contact

4.1 Constitutive Model

For volumetric elastic objects, we adopt the fixed corotational model from [SHS12], though any hyperelastic potential may be used. With this choice, the stress satisfies

$$\begin{aligned} \psi(\mathbf{F}) &= \mu \sum_i (\sigma_i - 1)^2 + \frac{\lambda}{2}(J-1)^2, \\ \mathbf{P} &= \mu(\mathbf{F} - \mathbf{R}) + \lambda(J-1)J\mathbf{F}^{-T}. \end{aligned} \tag{4.1}$$

Here μ and λ are the Lamé coefficients that express the material resistance for deformation and volume change, and σ_i are the singular values of the deformation gradient $\mathbf{F}$ computed according to the polar SVD convention of [ITF04] to allow for extreme deformation.

4.2 Discretization

Our hybrid FEM/MPM discretization of hyperelastic volumetric objects closely resembles that of traditional FEM for hyperelasticity [SB12]. However, our approach is largely motivated by the the MPM treatment of volumetric objects from Jiang et al. [JSS15] and Zhu et al. [ZZL17]. This method was originally designed to prevent the numerical fracture that would occur with volumetric objects in traditional particle-based MPM. We first discuss this approach and how it resolves self collision, followed by its drawbacks.

In this formulation, the state at time t^n consists of particles with positions $\mathbf{x}_p^n$ connected with a tetrahedron mesh with elements indexed by e, as in Lagrangian FEM. Furthermore, particles store velocities $\mathbf{v}_p^n$ and masses m_p. The MPM time step from time t^n to t^{n+1} consists of three steps: (1) mass (m_p) and momentum ($m_p\mathbf{v}_p^n$) are transferred from particles to the grid using weights ($w_{\mathbf{i}p}^n = N(\mathbf{x}_p^n - \mathbf{x}_\mathbf{i})$) that describe the degree of interaction between particle p and grid node $\mathbf{i}$ and which are defined by Eulerian grid interpolation functions $N(\mathbf{x})$, (2) the grid momentum ($m_\mathbf{i}^n\mathbf{v}_\mathbf{i}^n$) is updated in a variational way from the potential energy in the system and finally, (3) the motion of the grid under the updated momentum is interpolated to the particles. The process of updating the grid momentum in step (2) uses the updated Lagrangian convention where the time t^n configuration serves as the reference, rather than the $t = 0$ configuration in a Lagrangian discretization. With this updated Lagrangian convention, the particles $\mathbf{x}_p^n$ are moved by the grid via interpolation $\mathbf{x}_p^{n+1} = \sum_\mathbf{i} \mathbf{x}_\mathbf{i}^{n+1} w_{\mathbf{i}p}^n$, and they change the potential energy via the per-element deformation gradient computed as in standard FEM (see Equation (4.2)). The grid node vertices $\mathbf{x}_\mathbf{i}$, which are allowed to move temporarily as $\mathbf{x}_\mathbf{i}^{n+1} = \mathbf{x}_\mathbf{i} + \Delta t \mathbf{v}_\mathbf{i}^{n+1}$, serve as degrees of freedom. When the spatial discretization is done variationally from the potential energy, this step is almost identically what is done in a Lagrangian FEM discretization of elastoplasticity [SB12]. In this sense, the method can be interpreted as continually remeshing the domain of the material, where the transfer process in step (1) is all that is needed to define the mesh at a given time step. We refer the reader to [JSS15, JST16] for more basic MPM details.

The MPM update only considers the variation of the potential energy with respect to grid degrees of freedom; nothing explicit is done to model self collision. Self collision is modeled as if it were an elastic phenomenon, and by virtue of switching between particle and grid representations. We describe these two aspects of collision resolution as **type (i)** and **type (ii)**.

Type (i): The grid transfers in step (1) ultimately remesh the domain (see Figure 4.1). By transferring to the grid, and using an updated Lagrangian formulation where the grid nodes are updated based on the variation of the potential energy in Equation (4.1), MPM

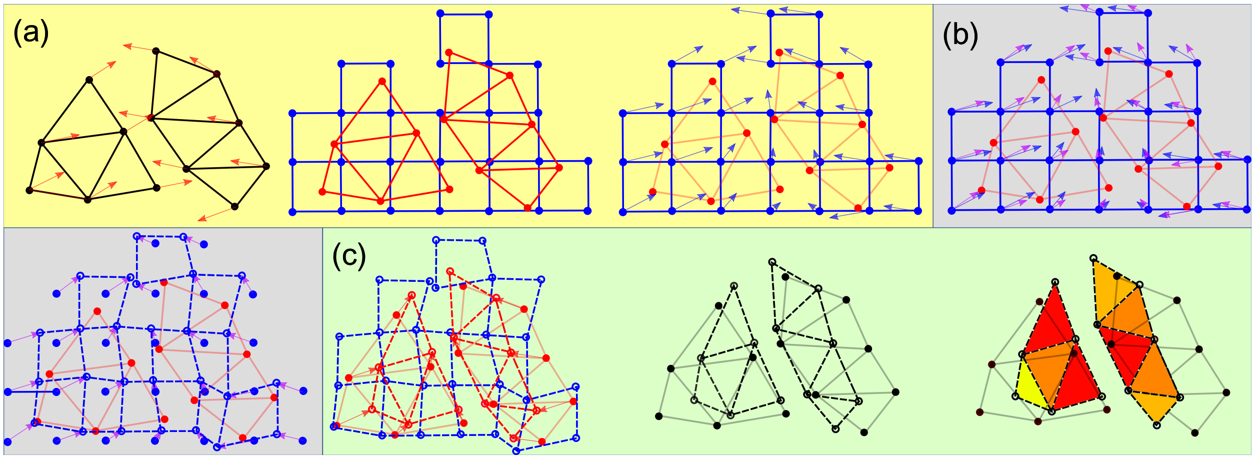

Figure 4.1: **MPM Overview**. The steps in the MPM update are: (a) The Lagrangian quantities (black and red) are transferred to an Eulerian grid (blue), which may be viewed as a new FEM mesh. (b) Grid nodes receive new velocities (purple) from updated Lagrangian elastic updates and are temporarily moved with those velocities. (c) The Lagrangian quantities are updated by interpolating from the new positions and velocities of the Eulerian grid nodes. The triangles are colored based on the amount of compression.

essentially uses a new FEM mesh (blue in Figure 4.1) to calculate the elastic update. This process creates new connections in the updated Lagrangian mesh and once they are made, collision inducing modes are penalized via the potential energy in the system (see Figure 4.1). For example, collision trajectories in the particles will induce compression in elements of the Eulerian grid which would be penalized from the elastic potential in the system.

Type (ii): Since the motion of the Eulerian grid after the momentum update in step (2) is interpolated to the particles using continuous interpolating functions, particle collisions cannot occur as long as the Eulerian mesh is not tangled by the motion. This can be guaranteed with a CFL restriction. Collisions occur because of discontinuities in the velocity, e.g. consider two particles next to each other with opposing velocities. Transferring to and from the grid smooths the particle velocities, which ultimately prevents collision. In fact, an updated Lagrangian MPM simulation with no constitutive model on the particles at all can still prevent material collision, simply by virtue of the **type (ii)** interactions (see Figure 4.2).

These modes of collision resolution are simplistic, but limited by several drawbacks. For volumetric objects, the **type (i)** interactions are unable to regulate the potential energy with a plasticity model derived from Coulomb friction as in [JGT17] and Chapter 3. The mesh is volumetric and therefore does not have the flexibility of codimension that can be used to model contact through the continuum. There are no directions left for plastic flow of the type designed in [JGT17] that could be used to satisfy the Coulomb friction stress constraints. This can lead to unregulated resistance to shearing and cohesion as the elastic potential will still increase with these modes, even though that is not consistent with Coulomb friction (see Figure 4.3). Furthermore, the updated Lagrangian treatment of the stress-based momentum leads to visual interaction at a distance and persistent wrinkling when the grid resolution is too low [JSS15, FGG17b, HN17]. Additionally, when the grid resolution is too high, **type (i)** and **type (ii)** interactions have no effect and the method does not prevent collision (see Figure 4.2). It is therefore required that for volumetric elasticity, the Lagrangian mesh resolution must be about the same as the Eulerian grid resolution. This is suboptimal when a coarse Lagrangian mesh suffices for resolution of deformation modes since collision interactions will also be resolved at a coarse scale with visible separation when the Eulerian grid resolution is set appropriately.

4.2.1 Hybrid Lagrangian MPM for elastic solids

Our method is designed by abandoning the **type (i)** collision prevention for volumetric meshes and the updated Lagrangian integration of the elastic forces in general. Instead we use a splitting approach where elastic forces are applied in a Lagrangian way, and **type (ii)** interactions are integrated by MPM with no elastic force computation. We achieve this by introducing collision particles $\mathbf{x}_q^n$ which are sampled on the boundary of the volumetric elastic mesh. These particles are not true degrees of freedom and are tied to the mesh during the Lagrangian update. They are then used to generate **type (ii)** collision prevention. We show that their response defines a type of impulse that can be regulated by Coulomb friction and applied to the mesh at the end of the time step. Furthermore, because the collision particles can be sampled at a density proportional to the grid spacing, we show that they remove the effect of grid resolution on collision resolution (see Figure 4.4).

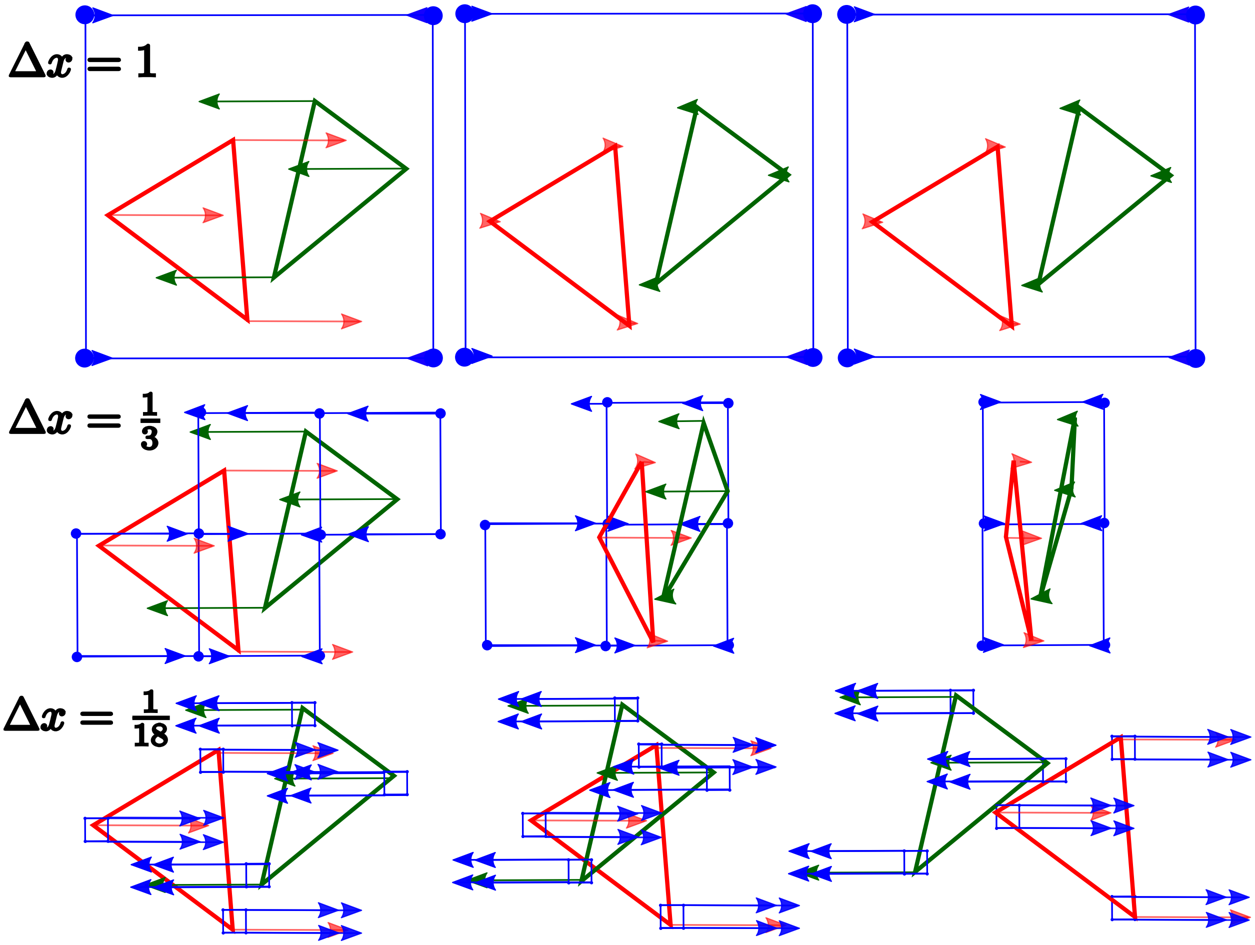

Figure 4.2: **Type (ii) interations with different Δx.** At appropiate grid resolution (middle row), MPM prevents material collision even without constitutive model. However, when the grid resolution is too low (top row), objects are separated at a distance, and when the grid resolution is too high (bottom row), the MPM grids may miss a collision.

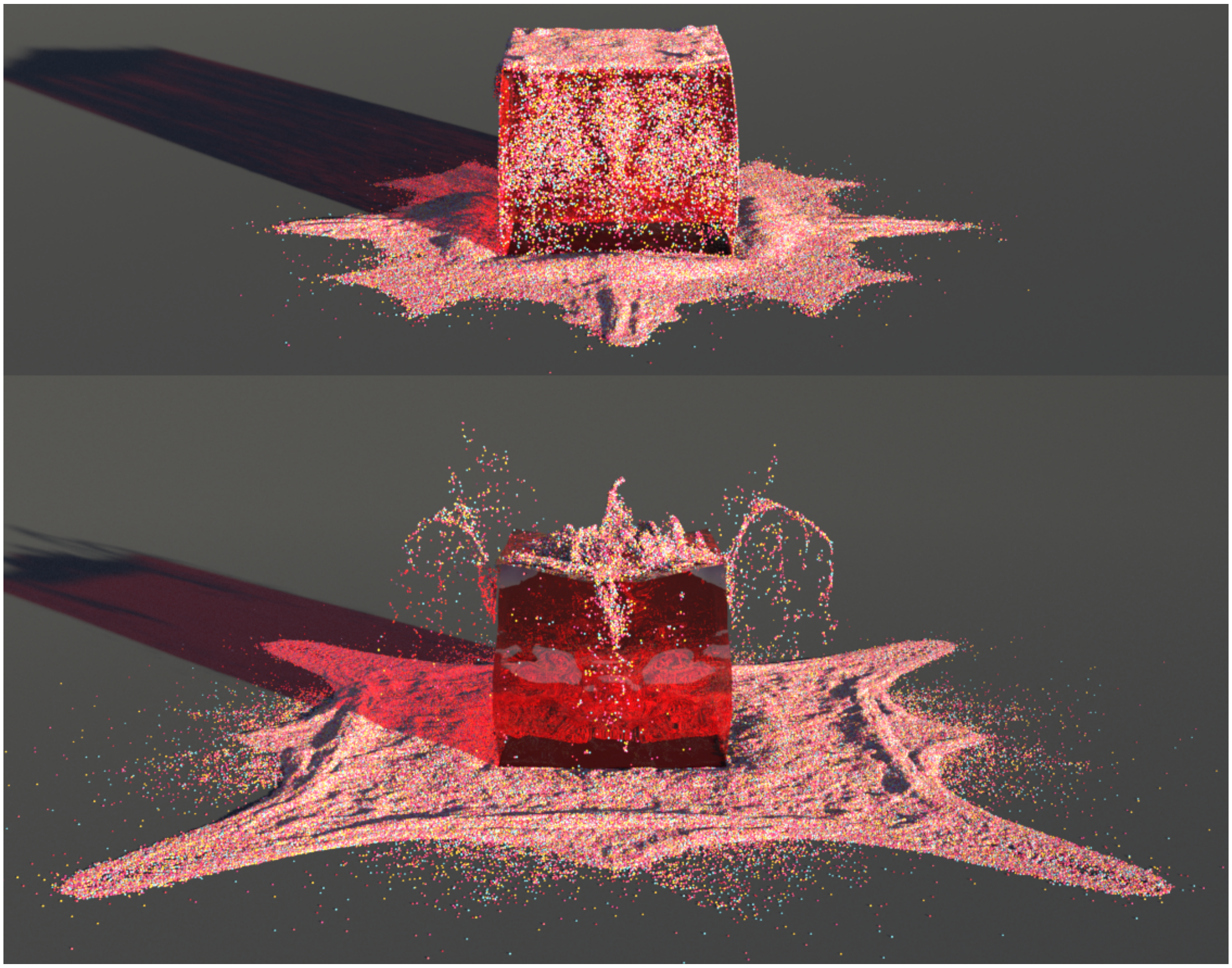

Figure 4.3: Comparison between MPM (top) and our method (bottom).

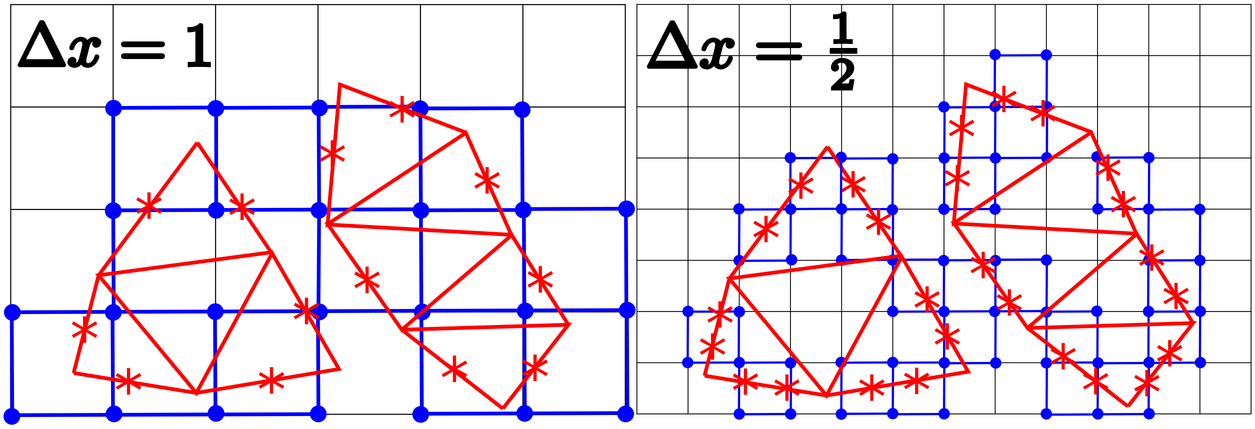

Figure 4.4: **Collision particles**. Sampling density based on Eulerian grid Δx.

Our approach uses the same discrete state as in [JSS15]: time t^n, particle positions $\mathbf{x}_p^n$ connected with a tetrahedron mesh, velocities $\mathbf{v}_p^n$, and masses m_p. In addition, we store the collision particles $\mathbf{x}_q^n$ sampled on the boundary of the tetrahedron mesh. We summarize essential steps in the algorithm for updating our discrete state to time t^{n+1} below. Note the difference between our method and traditional MPM steps sketched in Section §1.1.

1. **Lagrangian update:** Update particle velocities from potential-energy-based and body forces, and interpolate velocities to collision particles. §4.2.2

2. **Transfer to grid:** Transfer mass and momentum from collision particles to grid. §4.2.3.1

3. **Transfer to collision particles**: Transfer velocities from grid back to collision particles. §4.2.3.2

4. **Apply impulses:** Calculate the impulse applied to each boundary mesh using the velocity change in collision particles and update velocities of particles on the boundary mesh.

5. **Update positions:** Update particle positions and elastic states. §4.2.5.

4.2.2 Lagrangian update

We consider the case of piecewise linear interpolation over a tetrahedron mesh. The deformation gradient varies in a piecewise constant manner with each element, which we denote as $\mathbf{F}_e$. With this convention, the FEM force per particle $\mathbf{f}_p$ can be seen as the negative gradient of the the total potential energy Ψ with respect to grid node positions:

$$\mathbf{F}_e(\mathbf{x}) = \sum_p \mathbf{x}_p \frac{\partial \tilde{N}_p}{\partial \mathbf{X}}(\mathbf{X}_e) \tag{4.2}$$

$$\Psi(\mathbf{x}) = \sum_e \psi(\mathbf{F}_e(\mathbf{x}))V_e^0 \tag{4.3}$$

$$\mathbf{f}_p(\mathbf{x}) = -\sum_e \frac{\partial \psi}{\partial \mathbf{F}}(\mathbf{F}_e(\mathbf{x})) : \frac{\partial \mathbf{F}_e}{\partial \mathbf{x}_p}(\mathbf{x})V_e^0 \tag{4.4}$$

$$= -\sum_e \mathbf{P}(\mathbf{F}_e(\mathbf{x}))\frac{\partial \tilde{N}_p}{\partial \mathbf{X}}V_e^0. \tag{4.5}$$

Here $\mathbf{x} \in \mathbb{R}^{3n_p}$ refers to the vector of all particles $\mathbf{x}_p$, where n_p is the total number of particles, Ψ is the total potential energy which is a sum of tetrahedron element contributions $\psi(\mathbf{F}_e)V_e^0$, where ψ is the potential energy density in Equation (4.1), V_e^0 is the volume of the element in the initial state, $\tilde{N}_p$ is the piecewise linear interpolating function associated with particle $\mathbf{x}_p$, and $\mathbf{X}_e$ is the tetrahedron barycenter in the time $t = 0$ configuration. We refer the reader to Sifakis and Barbic [SB12] for a more detailed derivation.

The FEM update uses the usual Lagrangian view of the governing physics. The internal force is the negative gradient of the potential energy in Equation (4.5). Particle velocities are updated according to forces computed at particle positions $\mathbf{x}_p^{n+\alpha}$, where symplectic Euler integration corresponds to $\alpha = 0$ and backward Euler corresponds to $\alpha = 1$:

$$\mathbf{v}_p^* = \mathbf{v}_p^n + \Delta t \frac{\mathbf{f}_p(\mathbf{x}^{n+\alpha})}{m_p}. \tag{4.6}$$

When damping is required while using symplectic Euler integration, we construct a background Eulerian grid with Δx comparable to the mesh size and transfer the vecloty to and then back from the grid using APIC with RPIC damping as described in [JGT17]. We can

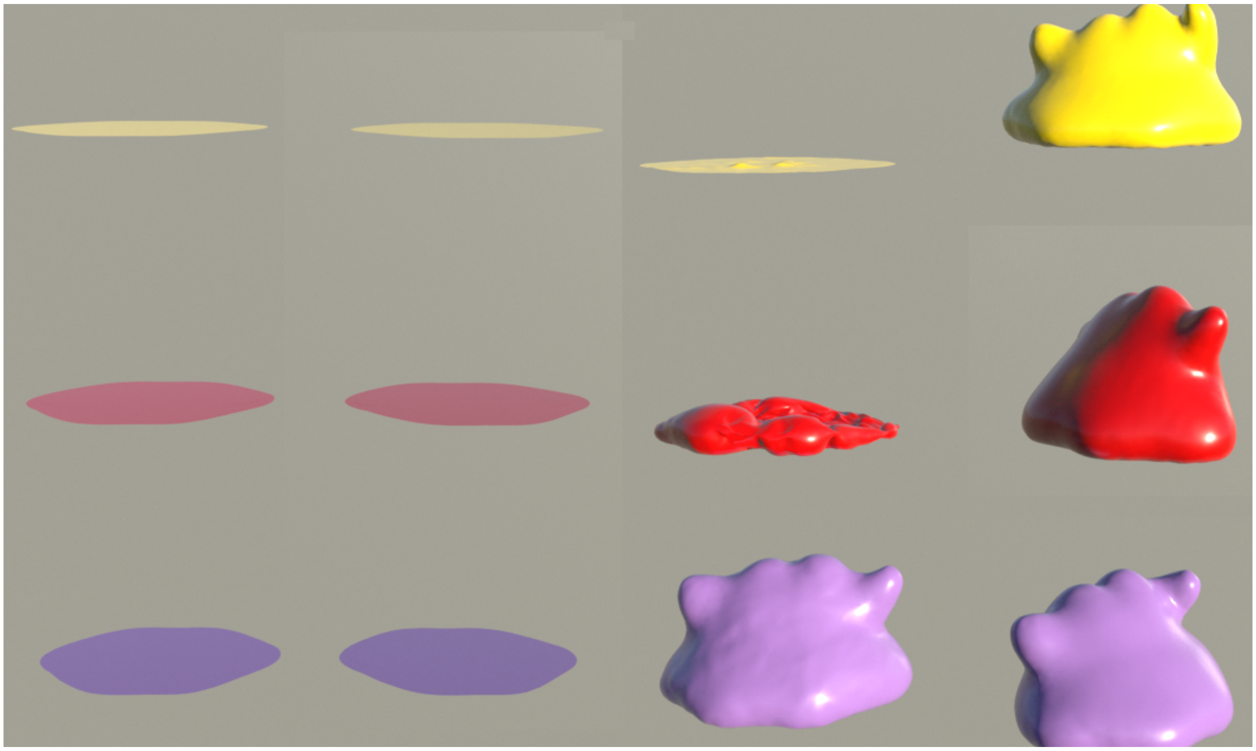

Figure 4.5: **Element inversion**. MPM (left) has difficulties when elements invert, especially with low grid resolution (yellow and red). Our method (right) handles element inversions with ease.

even perform the transfers multiple times when more damping is desired. For interior particles, $\mathbf{v}_p^{n+1} = \mathbf{v}_p^*$. On the other hand, for particles on the boundary mesh, we interpolate their velocities and positions to collision particles using

$$\mathbf{v}_q^* = \sum_p b_{pq} \mathbf{v}_p^* \tag{4.7}$$

$$\mathbf{x}_q^n = \sum_p b_{pq} \mathbf{x}_p^n \tag{4.8}$$

where b_{pq} is the barycentric weight of the point q relative to p. We also assign to each point q an outward normal vector $\mathbf{n}_q$ inherited from the face of the mesh that q is tied to.

4.2.3 Grid transfers

4.2.3.1 Particle to Grid

To process collision and contact, we transfer mass and momentum from collision particles $\mathbf{x}_q^n$ to grid nodes $\mathbf{x}_\mathbf{i}$ using standard MPM transfers

$$m_\mathbf{i}^n = \sum_q w_{\mathbf{i}q}^n m_q \tag{4.9}$$

$$\mathbf{v}_\mathbf{i}^* = \frac{1}{m_\mathbf{i}^n} \sum_q w_{\mathbf{i}q}^n m_q \mathbf{v}_q^*. \tag{4.10}$$

Here $w_{\mathbf{i}q}^n = N(\mathbf{x}_q^n - \mathbf{x}_\mathbf{i})$ is the weight of interaction between particle $\mathbf{x}_q^n$ and grid node $\mathbf{x}_\mathbf{i}$, as in standard MPM.

4.2.3.2 Grid to particle

Without any constitutive model on the grid, we proceed directly to the grid to particle step. The grid to particle transfer defines the velocity local to collision particle $\mathbf{x}_q^n$ in terms of $\mathbf{v}_q^\star$ from

$$\mathbf{v}_q^\star = \sum_\mathbf{i} w_{\mathbf{i}q}^n \mathbf{v}_\mathbf{i}^*. \tag{4.11}$$

4.2.4 Apply impulse

Since the velocity $\mathbf{v}_q^\star$ is interpolated from an updated Lagrangian background grid, the boundary of the mesh is safe from self-intersection if it is moved with $\mathbf{v}_q^\star$. However, the change may not be consistent with a Coulomb friction interaction, and the response can even be cohesive. In the case of a cohesive response after collision, we reject the change.

That is, when

$$\mathbf{v}_r = \mathbf{v}_q^{\star} - \mathbf{v}_q^{*} \tag{4.12}$$

$$\mathbf{v}_r \cdot \mathbf{n}_q \geq 0 \tag{4.13}$$

the updated Lagrangian mesh detects a separation instead of collision, and the collision particle keeps the velocity from the FEM update $\mathbf{v}_q^*$. On the other hand, if

$$\mathbf{v}_r \cdot \mathbf{n}_q < 0 \tag{4.14}$$

we apply an elastic impulse $I_q\mathbf{n}_q$ to the mesh at position $\mathbf{x}_q^n$ where $I_q = 2m_q\mathbf{v}_r \cdot \mathbf{n}_q$. We also allow for friction using Coulomb's model with the friction parameter μ. When an elastic impulse of magnitude I_q would be applied based on condition (4.14), Coulomb friction admits a change in magnitude of tangential velocity of at most $-\mu\frac{I_q}{m_q}$. So the combined velocity change on collision particle q is then

$$\Delta\mathbf{v}_q = \frac{I_q\mathbf{n}_q}{m_q} + \min\left(\|\mathbf{v}_t\|, -\mu\frac{I_q}{m_q}\right)\frac{\mathbf{v}_t}{\|\mathbf{v}_t\|}, \tag{4.15}$$

where $\mathbf{v}_t = \mathbf{v}_r - \mathbf{v}_r \cdot \mathbf{n}_q\mathbf{n}_q$. We then transfer this change to the particles p as

$$\Delta\mathbf{v}_p = \mathbf{v}_p^{n+1} - \mathbf{v}_p^* = \sum_q \tilde{b}_{pq}\Delta\mathbf{v}_q \tag{4.16}$$

where

$$\tilde{b}_{pq} = \frac{b_{pq}m_q}{\sum_r b_{pr}m_r} \tag{4.17}$$

are the normalized weights defined from the barycentric weights used to transfer from particles to collision particles.

4.2.5 Update positions and elastic state

For boundary particles, we adopt symplectic Euler time integration

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^n + \Delta\mathbf{v}_p \tag{4.18}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t\mathbf{v}_p^{n+1} \tag{4.19}$$

For interior particles, the update is in accordance with either symplectic Euler or backward Euler, depending on the choice of α in Equation (4.6):

$$\mathbf{v}_p^{n+1} = \mathbf{v}_p^* \tag{4.20}$$

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t\mathbf{v}_p^{n+1}. \tag{4.21}$$

4.3 Rigid Body

Two-way rigid body coupling may be achieved with a treatment similar to volumetric elastic objects. We sample collision particles on the boundary in the same fashion as in Section 4.2.1 and then uniformly distribute the mass of the rigid body to the collision particles. However, we found that unlike for volumetric elastic objects, **type (ii)** interactions on the grid alone are not enough to resolve collisions. Instead we endow the collision particles with the potential described in [JGT17] and Chapter 3 to penalize contact. Specifically, we update the deformation gradient $\mathbf{F}_q$ from time t_n to t_{n+1} in the following way. Let $\mathbf{x}_\alpha$ and $\mathbf{X}_\alpha$, $\alpha \in \{0, 1, 2\}$ be the current and initial positions of the vertices of the triangle that collision particle q is tied to. Let $\bar{\mathbf{D}}\mathbf{D}_{q,\beta} = \mathbf{X}_\beta - \mathbf{X}_0$ be the undeformed mesh element edge vectors (where $\beta = 1,\ 2$), and $\hat{\mathbf{d}}_{q,\beta}^E = \mathbf{x}_\beta^n - \mathbf{x}_0^n$ be the deformed edge vectors. We choose each $\bar{\mathbf{D}}\mathbf{D}_3$ to be unit-length and normal to $\bar{\mathbf{D}}\mathbf{D}_1$ and $\bar{\mathbf{D}}\mathbf{D}_2$, and evolve each one as in traditional MPM via $\hat{\mathbf{d}}_{q,3}^E = \boldsymbol{\nabla}\mathbf{x}_q\mathbf{d}_{q,3}^E$. Then $\hat{\mathbf{F}}_q^E = \hat{\mathbf{d}}_q^E\bar{\mathbf{D}}\mathbf{D}_q^{-1}$. Following [JGT17] and Chapter 3, we let $\hat{\mathbf{F}}_q^E = \mathbf{Q}\hat{\mathbf{R}}$

be the QR decomposition of $\hat{\mathbf{F}}_q^E$ and design a collision energy density $\psi(\hat{\mathbf{R}}) = f(\hat{\mathbf{R}}) + g(\hat{\mathbf{R}})$,

$$f(\hat{\mathbf{R}}) = \begin{cases} \frac{k^c}{3}(1-\hat{r}_{33})^3 & 0 \leq \hat{r}_{33} \leq 1 \\ 0 & \hat{r}_{33} > 1 \end{cases}, \quad g(\hat{\mathbf{R}}) = \frac{\gamma}{2}(\hat{r}_{13}^2 + \hat{r}_{23}^2) \tag{4.22}$$

where $\hat{r}_{ij}$ is the ij-th entry of $\hat{\mathbf{R}}$. We resolve the force which is the negative derivative of this energy on the MPM background grid, and we refer the reader to [JSS15, JST16] for more details. Plasticity is then applied according to [JGT17] and Chapter 3 to give $\mathbf{R}$

$$r_{33} = \begin{cases} \hat{r}_{33} & 0 < \hat{r}_{33} \leq 1 \\ 1 & \hat{r}_{33} > 1 \end{cases}, \quad r_{\beta 3} = h(\hat{r}_{13}, \hat{r}_{23}, r_{33})\hat{r}_{\beta 3} \tag{4.23}$$

$$h(\hat{r}_{13}, \hat{r}_{23}, r_{33}) = \min\left(1, \frac{c_F k^c \left(1 - r_{33}\right)^2}{\gamma\sqrt{\hat{r}_{13}^2 + \hat{r}_{23}^2}}\right) \tag{4.24}$$

Finally, we update the deformation gradient with $\mathbf{F}_q^{n+1} = \mathbf{QR}$.

Let $\mathbf{v}_q^* = \sum_{\mathbf{i}} w_{\mathbf{i}q}^n \mathbf{v}_{\mathbf{i}}^*$, where $\mathbf{v}_{\mathbf{i}}^*$ is the grid velocity after the MPM force update, and let $\mathbf{v}_r = \mathbf{v}_q^* - \mathbf{v}_q$. If $\mathbf{v}_r \cdot \mathbf{n}_q < 0$, we apply an impulse $\mathbf{I}_q$ to the rigid bodies to update velocity $\mathbf{v}$ and angular velocity $\boldsymbol{\omega}$ via

$$I_q = m_q \mathbf{v}_r \cdot \mathbf{n}_q \tag{4.25}$$

$$\mathbf{v}_t = \mathbf{v}_r - \mathbf{v}_r \cdot \mathbf{n}_q \mathbf{n}_q \tag{4.26}$$

$$\mathbf{I}_q = I_q \mathbf{n}_q + m_q \min\left(\|\mathbf{v}_t\|, -\mu \frac{I_q}{m_q}\right) \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|} \tag{4.27}$$

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \sum_q \frac{\mathbf{I}_q}{m_q} \tag{4.28}$$

$$\boldsymbol{\omega}^{n+1} = \boldsymbol{\omega}^n + \sum_q \mathbf{I}^{-1}(\mathbf{r} \times \mathbf{I}_q) \tag{4.29}$$

where $\mathbf{r}$ is the vector from the rigid body's center of mass to the application point of the impulse, and $\mathbf{I}$ is the inertia tensor.

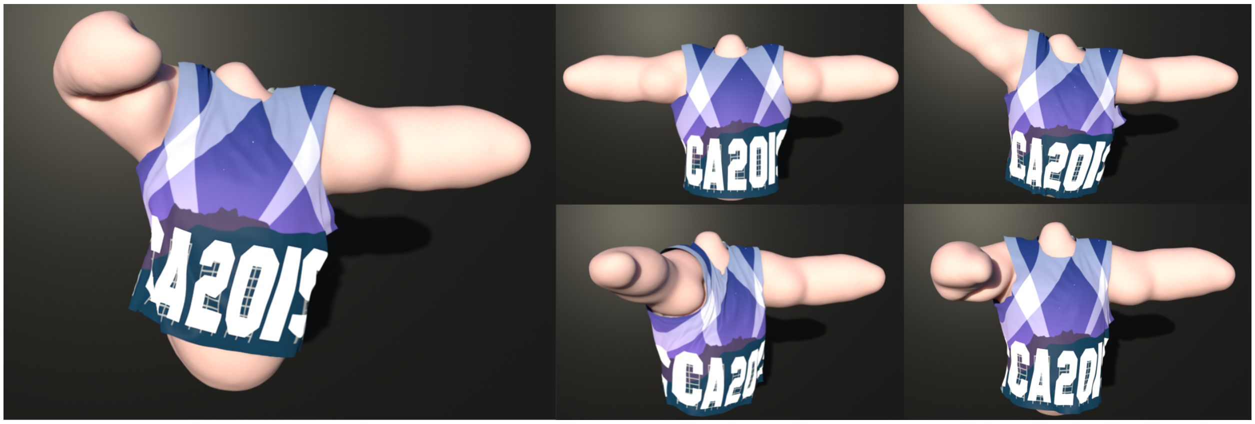

Figure 4.6: **Skin and shirt**. The skin of a mannequin is coupled with clothing simulated with MPM.

4.4 Coupling with traditional MPM

Our method easily couples with traditional MPM particles such as snow, sand, and clothing through the Eulerian background grid. To prevent numerical cohesion between phases common to MPM, we adopt separate grids for volumetric objects and MPM particles. The interactions among them are resolved by inelastic collision impulses between collocated grid nodes from different grids. The contact normal $\mathbf{n}_i^n$ is determined by the grid averaged normal of collision particles, which takes the form:

$$\mathbf{n}_i^n = \frac{\sum_q w_{\mathbf{i}q}\mathbf{n}_q}{\|\sum_q w_{\mathbf{i}q}\mathbf{n}_q\|} \tag{4.30}$$

We provide implementation details on cohesion-free coupling with traditional MPM in Appendix §C.1.

4.5 Results

We demonstrate the efficacy of our method with a number of representative examples that illustrate the dynamics of volumetric objects, and show that our method couples with granular materials, clothing and rigid bodies. We list the runtime performance for our examples in Table 4.1. All simulations were run on an Intel Xeon E5-2690 V2 system with 20 threads

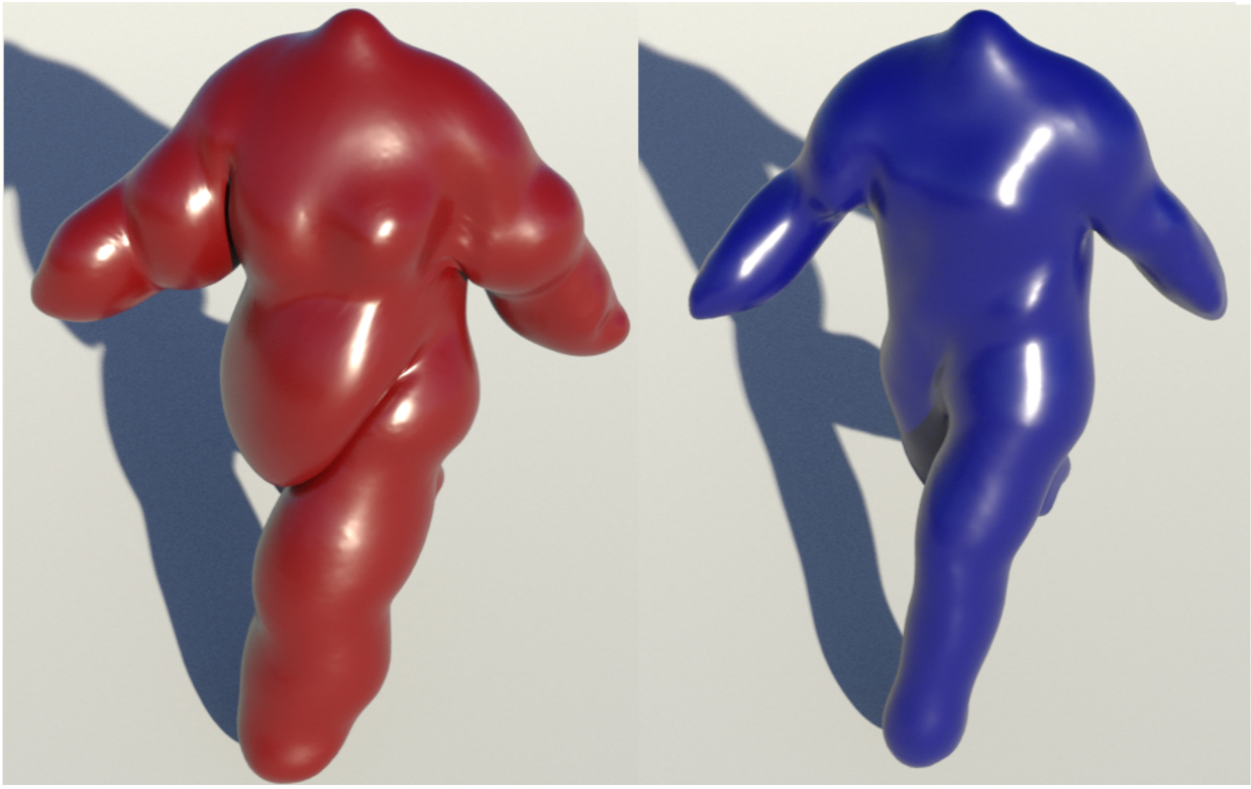

Figure 4.7: **Walking mannequins**. Our method handles the numerous collisions occurring in the scene with walking characters.

and 128GB of RAM. We report the timing in terms of average seconds of computation per frame. We chose Δt in an adaptive manner that is restricted by a CFL condition when the particle velocities are high, i.e., we do not allow particles to move further than the CFL number times Δx in a time step.

4.5.1 Volumetric objects

We demonstrate the robustness of our method for resolving collisions between volumetric objects. We demonstrate that our method correctly resolves frictional sliding without artifacts. In Figure 4.7, we show a skin simulation with walking characters in various body shapes. In Figure 4.3, we compare our approach with updated Lagrangian MPM, which exhibits excessive cohesion and numerical friction. We also show that our method removes the requirement of comparable grid and mesh resolution. We use a moderate resolution Lagrangian mesh to resolve the dynamics of the bunnies and Jell-O's and a high resolution Eulerian grid to resolve more detailed behaviors of the sand. In contrast, updated Lagrangian MPM would require a high resolution Lagrangian mesh for bunnies and Jell-O's in order to resolve collisions between phases. Our method handles extreme deformation and even element inversion as demostrated in Figure 4.5. MPM fails to recover the original shape of the object when the grid resolution is low and exhibits high frequency noise when the grid resolution is high. On the other hand, the elastic object recovers its original shape with any grid resolutions using our method.

4.5.2 Coupling with MPM and rigid bodies

Our method also supports coupling with rigid bodies as well as traditional MPM particles such as snow, sand and clothing. In Figure 4.6, we demostrate the coupling of soft tissues with clothing material simulated with MPM as in [JGT17]. In Figure 4.3, colored sand is poured on top of three Jell-O, generating interesting patterns. In Figure 4.8, elastic dittos and a column of sand are poured on a series of pinwheels simulated as rigid bodies, setting them in motion.

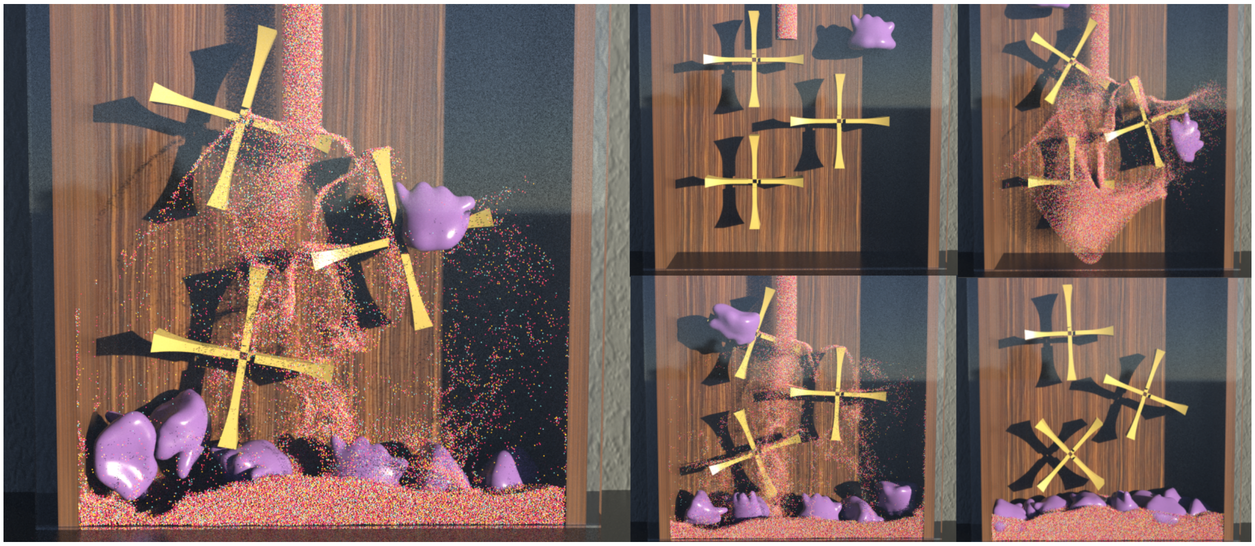

Figure 4.8: **Rigid body coupling**. Elastic volumetric objects are coupled with MPM particles and rigid bodies in a scene with intense collisions.

	Time	Element #	Particle #	Δx	CFL
Mannequin (Fig. 4.7 left)	39	933K	41k	0.05	0.6
Mannequin (Fig. 4.7 right)	27	641K	31K	0.05	0.6
Pinwheel (Fig. 4.8)	89	93K	930K	0.5	0.6
Jell-O (MPM) (Fig. 4.3 top)	73	8.64M	5.41M	0.005	0.6
Jell-O (Hybrid) (Fig. 4.3 bottom)	220	1.08M	3.81M	0.005	0.6
Bunnies (MPM) (Fig. 1.2 left)	186	3.97M	2.67M	0.1	0.6
Bunnies (Hybrid) (Fig. 1.2 right)	66	201K	1.99M	0.1	0.6
Skin and shirt (Fig. 4.6)	3	207K	120K	0.006	0.6

Table 4.1: All simulations were run on Intel Xeon E5-2690 v4 system with 20 threads and 128GB of RAM. Simulation time is measure in seconds per frame. Element # denotes number of segments for number of tetrahedra for volumetric simulations. Particle # denotes the total number of MPM particles and collision particles where applicable.

4.6 Discussion and Limitations

While our approaches address many shortcomings in existing techniques, there are a number of limitations that persist. While our treatment of rigid body dynamics is useful for coupling with elastoplastic materials like sand, soft tissues, etc., our approach is not ideally suited for interactions between rigid bodies. Our approach fails to resolve simple cases like stacking of a few rigid bodies without penetration and/or grid based separation artifacts. Also, our collision impulses do not provide any geometric guarantees against self collision, as in e.g. [BFA02]. If large time steps are taken, material will interpenetrate. In general this can be avoided by obeying a CFL condition, as is generally true with MPM.

APPENDIX A

Polynomial Particle-In-Cell Method

A.1 List of Bases

A.1.1 Linear interpolation

Polynomials of the form

$$s(\mathbf{z}) = \prod_{\beta=1}^{d} z_\beta^{i_\beta}, \quad i_\beta = 0, 1$$

are all we need for linear interpolation. We have $\mathcal{S}_{pr}^n \cdot \left(\mathbf{m}_p^n \mathcal{S}_{pt}^n\right) = 0$ for all $r \neq t$.

A.1.2 Quadratic interpolation

For quadratic interpolation, by replacing $z_\beta^{i_\beta}$ with $g_\beta(w) = w^2 - \frac{x_{p\beta}^n\left(\Delta x^2 - 4(x_{p\beta}^n)^2\right)}{\Delta x^2} w - \frac{\Delta x^2}{4}$ in

$$s(\mathbf{z}) = \prod_{\beta=1}^{d} z_\beta^{i_\beta}, \quad i_\beta = 0, 1, 2$$

whenever $i_\beta = 2$, we get the full set of basis vectors. For completeness we list all the bases below.

In 2D,

$$
\begin{aligned}
&s_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = 1 && \\
&s_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = x_{\mathbf{i}_{kp1}} - x^n_{p1} && s_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = x_{\mathbf{i}_{kp2}} - x^n_{p2} \\
&s_4(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = (x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2}) && \\
&s_5(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1}) && s_6(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2}) \\
&s_7(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2}) && s_8(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2})(x_{\mathbf{i}_{kp1}} - x^n_{p1}) \\
&s_9(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2}) &&
\end{aligned}
$$

In 3D,

$$
\begin{aligned}
&s_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = 1 \\
&s_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = x_{\mathbf{i}_{kp1}} - x^n_{p1} \\
&s_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = x_{\mathbf{i}_{kp2}} - x^n_{p2} \\
&s_4(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = x_{\mathbf{i}_{kp3}} - x^n_{p3} \\
&s_5(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = (x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2}) \\
&s_6(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = (x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_7(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = (x_{\mathbf{i}_{kp2}} - x^n_{p2})(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_8(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = (x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2})(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_9(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1}) \\
&s_{10}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2}) \\
&s_{11}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_3(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_{12}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2}) \\
&s_{13}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2})g_3(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_{14}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})g_3(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_{15}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})g_2(x_{\mathbf{i}_{kp2}} - x^n_{p2})g_3(x_{\mathbf{i}_{kp3}} - x^n_{p3})
\end{aligned}
$$

$$
\begin{aligned}
&s_{16}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2}) \\
&s_{17}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp3}} - x^n_{p3}) \\
&s_{18}(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = g_1(x_{\mathbf{i}_{kp1}} - x^n_{p1})(x_{\mathbf{i}_{kp2}} - x^n_{p2})(x_{\mathbf{i}_{kp3}} - x^n_{p3})
\end{aligned}
$$

$$s_{19}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_2(x_{\mathbf{i}_{kp2}} - x_{p2}^n)(x_{\mathbf{i}_{kp1}} - x_{p1}^n)$$

$$s_{20}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_2(x_{\mathbf{i}_{kp2}} - x_{p2}^n)(x_{\mathbf{i}_{kp3}} - x_{p3}^n)$$

$$s_{21}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_2(x_{\mathbf{i}_{kp2}} - x_{p2}^n)(x_{\mathbf{i}_{kp1}} - x_{p1}^n)(x_{\mathbf{i}_{kp3}} - x_{p3}^n)$$

$$s_{22}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_3(x_{\mathbf{i}_{kp3}} - x_{p3}^n)(x_{\mathbf{i}_{kp1}} - x_{p1}^n)$$

$$s_{23}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_3(x_{\mathbf{i}_{kp3}} - x_{p3}^n)(x_{\mathbf{i}_{kp2}} - x_{p2}^n)$$

$$s_{24}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_3(x_{\mathbf{i}_{kp3}} - x_{p3}^n)(x_{\mathbf{i}_{kp1}} - x_{p1}^n)(x_{\mathbf{i}_{kp2}} - x_{p2}^n)$$

$$s_{25}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_1(x_{\mathbf{i}_{kp1}} - x_{p1}^n)g_2(x_{\mathbf{i}_{kp2}} - x_{p2}^n)(x_{\mathbf{i}_{kp3}} - x_{p3}^n)$$

$$s_{26}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_1(x_{\mathbf{i}_{kp1}} - x_{p1}^n)g_3(x_{\mathbf{i}_{kp3}} - x_{p3}^n)(x_{\mathbf{i}_{kp2}} - x_{p2}^n)$$

$$s_{27}(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n) = g_2(x_{\mathbf{i}_{kp2}} - x_{p2}^n)g_3(x_{\mathbf{i}_{kp3}} - x_{p3}^n)(x_{\mathbf{i}_{kp1}} - x_{p1}^n)$$

The entries for $\mathcal{S}_{pr}^n \cdot \left(\mathbf{m}_p^n \mathcal{S}_{pr}^n\right)$, $r = 1, 2, \cdots, 27$ are:

$$
\begin{aligned}
&1,\\
&\tfrac{\Delta x^2}{4}, \tfrac{\Delta x^2}{4}, \tfrac{\Delta x^2}{4},\\
&\tfrac{\Delta x^4}{16}, \tfrac{\Delta x^4}{16}, \tfrac{\Delta x^4}{16}, \tfrac{\Delta x^6}{64},\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)}{16\Delta x^2}, \tfrac{\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)}{16\Delta x^2}, \tfrac{\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{16\Delta x^2},\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)}{256\Delta x^4},\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{256\Delta x^4},\\
&\tfrac{\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{256\Delta x^4},\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{4096\Delta x^6},\\
&\tfrac{1}{64}\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right),\\
&\tfrac{1}{64}\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right),\\
&\tfrac{1}{256}\left(3\Delta x^2-4x^2\right)\left(\Delta x^3-4\Delta x x^2\right)^2,\\
&\tfrac{1}{64}\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right),\\
&\tfrac{1}{64}\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right),\\
&\tfrac{1}{256}\left(3\Delta x^2-4y^2\right)\left(\Delta x^3-4\Delta x y^2\right)^2,\\
&\tfrac{1}{64}\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right),\\
&\tfrac{1}{64}\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right),\\
&\tfrac{1}{256}\left(3\Delta x^2-4z^2\right)\left(\Delta x^3-4\Delta x z^2\right)^2,\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)}{1024\Delta x^2},\\
&\tfrac{\left(\Delta x^2-4x^2\right)^2\left(3\Delta x^2-4x^2\right)\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{1024\Delta x^2},\\
&\tfrac{\left(\Delta x^2-4y^2\right)^2\left(3\Delta x^2-4y^2\right)\left(\Delta x^2-4z^2\right)^2\left(3\Delta x^2-4z^2\right)}{1024\Delta x^2}
\end{aligned}
$$

A.2 Grid to Particle

From grid to particle, we wish to find **c** such that

$$
\begin{aligned}
\sum_{t=1}^{N_r} \mathcal{S}_{pr}^n \cdot \left(\mathbf{m}_p^n \mathcal{S}_{pt}^n\right) c_{pt\alpha}^{n+1} = \mathcal{Q}_{pr\alpha}^n \cdot \left(\mathbf{M}_p^n \hat{\mathcal{V}}_p^{n+1}\right)\\
= \sum_{k=1}^{(N_B+1)^d} m_{\mathbf{i}_{kp}^n p}^n s_r(\mathbf{x}_{\mathbf{i}_{kp}^n} - \mathbf{x}_p^n)\hat{v}_{\mathbf{i}_{kp}^n \alpha}^{n+1}.
\end{aligned}
$$

The bases we choose satisfy the property that $\mathcal{S}^n_{pr} \cdot \left(\mathbf{m}^n_p \mathcal{S}^n_{pt}\right) = 0$ for $r \neq t$. So we have

$$\begin{aligned}
\mathcal{S}^n_{pr} \cdot \left(\mathbf{m}^n_p \mathcal{S}^n_{pr}\right) c^{n+1}_{pr\alpha} &= \mathcal{Q}^n_{pr\alpha} \cdot \left(\mathbf{M}^n_p \hat{\mathcal{V}}^{n+1}_p\right) \\
&= \sum_{k=1}^{(N_B+1)^d} m^n_{\mathbf{i}^n_{kp}p} s_r(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}.
\end{aligned}$$

For linear interpolation, the grid to particle transfer is similar to APIC. For quadratic interpolation, $\mathcal{S}^n_{pr} \cdot \left(\mathbf{m}^n_p \mathcal{S}^n_{pr}\right)$ can be zero for some r when $x^n_{p\alpha} = \pm\frac{h}{2}, \pm\frac{\sqrt{3}}{2}h$. However, we can still find a meaningful expression for $\mathbf{c}$.

In 2D, $m^n_{\mathbf{i}^n_{kp}p} = m_p N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = m_p N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)$, $\mathbf{c}$ can be computed from the formula below:

$$\begin{aligned}
c_{p1\alpha} &= \sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha} \\
c_{p2\alpha} &= \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1}) \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^2}{4}} \\
c_{p3\alpha} &= \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2}) \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^2}{4}} \\
c_{p4\alpha} &= \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp2}} - x_{p2}) \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{16}} \\
c_{p5\alpha} &= \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{2\Delta x^2} \\
c_{p6\alpha} &= \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{2\Delta x^2} \\
c_{p7\alpha} &= \frac{2\sum_{k=1}^{(N_B+1)^d} (x_{\mathbf{i}_{kp1}} - x_{p1}) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\Delta x^4} \\
c_{p8\alpha} &= \frac{2\sum_{k=1}^{(N_B+1)^d} (x_{\mathbf{i}_{kp2}} - x_{p2}) N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\Delta x^4} \\
c_{p9\alpha} &= \sum_{k=1}^{(N_B+1)^d} \frac{1}{4\Delta x^4} (-2)^{(i_{kp1}-1) \bmod 2} (-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}
\end{aligned}$$

In 3D, $m^n_{\mathbf{i}^n_{kp}p} = m_p N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) = m_p N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)$, $\mathbf{c}$ can be computed from the formula below:

$$c_{p1\alpha} = \sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}$$

$$c_{p2\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^2}{4}}$$

$$c_{p3\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^2}{4}}$$

$$c_{p4\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp3}} - x_{p3})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^2}{4}}$$

$$c_{p5\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp2}} - x_{p2})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{16}}$$

$$c_{p6\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})(x_{\mathbf{i}^n_{kp3}} - x_{p3})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{16}}$$

$$c_{p7\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp3}} - x_{p3})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{16}}$$

$$c_{p8\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp2}} - x_{p2})(x_{\mathbf{i}^n_{kp3}} - x_{p3})\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^6}{64}}$$

$$c_{p9\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{2\Delta x^2}$$

$$c_{p10\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{2\Delta x^2}$$

$$c_{p11\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp3}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{2\Delta x^2}$$

$$c_{p12\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp1}-1) \bmod 2}(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{4\Delta x^4}$$

$$c_{p13\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp1}-1) \bmod 2}(-2)^{(i_{kp3}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{4\Delta x^4}$$

$$c_{p14\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(-2)^{(i_{kp2}-1) \bmod 2}(-2)^{(i_{kp3}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{4\Delta x^4}$$

$$c_{p15\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} (-2)^{(i_{kp1}-1) \bmod 2}(-2)^{(i_{kp2}-1) \bmod 2}(-2)^{(i_{kp3}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{8\Delta x^6}$$

$$c_{p16\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p17\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp3}} - x_{p3})(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p18\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})(x_{\mathbf{i}^n_{kp3}} - x_{p3})(-2)^{(i_{kp1}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^6}{8}}$$

$$c_{p19\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p20\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp3}} - x_{p3})(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p21\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp3}} - x_{p3})(-2)^{(i_{kp2}-1) \bmod 2} \hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^6}{8}}$$

$$c_{p22\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(-2)^{(i_{kp3}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p23\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})(-2)^{(i_{kp3}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^4}{2}}$$

$$c_{p24\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p) N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(x_{\mathbf{i}^n_{kp2}} - x_{p2})(-2)^{(i_{kp3}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\frac{\Delta x^6}{8}}$$

$$c_{p25\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_3(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp3}} - x_{p3})(-2)^{(i_{kp1}-1) \bmod 2}(-2)^{(i_{kp2}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\Delta x^6}$$

$$c_{p26\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_2(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp2}} - x_{p2})(-2)^{(i_{kp1}-1) \bmod 2}(-2)^{(i_{kp3}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\Delta x^6}$$

$$c_{p27\alpha} = \frac{\sum_{k=1}^{(N_B+1)^d} N_1(\mathbf{x}_{\mathbf{i}^n_{kp}} - \mathbf{x}^n_p)(x_{\mathbf{i}^n_{kp1}} - x_{p1})(-2)^{(i_{kp2}-1) \bmod 2}(-2)^{(i_{kp3}-1) \bmod 2}\hat{v}^{n+1}_{\mathbf{i}^n_{kp}\alpha}}{\Delta x^6}$$

A.3 PolyPIC is Lossless

In this section, we prove that if we use PolyPIC to transfer grid momentum to particle and then directly transfer back without advecting, we get the exact same grid momentum back. For simplicity, we prove it for the one particle case.

Given grid mass $\mathbf{M}^n_p$ and grid velocity $\mathcal{V}^n_p$, the $\mathbf{c}^{n+1}_p$ we find by using full-interpolation PolyPIC is given by

$$\mathbf{c}^{n+1}_p = (\mathbf{Q}^{n\,T}_P \mathbf{M}^n_p \mathbf{Q}^n_p)^{-1} \mathbf{Q}^{nT}_p \mathbf{M}^n_p \mathcal{V}^n_p.$$

We want to show that the momentum $\hat{\mathbf{p}}$ we get from $\hat{\mathbf{p}} = \mathbf{M}^n_p \mathbf{Q}^n_p \mathbf{c}^{n+1}_p$ is equal to the original momentum on the grid $\mathbf{M}^n_p \mathbf{v}_i$. The key obversation is that $\mathbf{Q}^n_p$ is invertible: $(\mathbf{Q}^n_p)^T \mathbf{M}^n_p \mathbf{Q}^n_p$ is full rank diagonal, which means that $\mathbf{Q}_p^n$ is also full rank and therefore invertible.

$$\begin{aligned}\hat{\mathbf{p}} &= \mathbf{M}_p^n\mathbf{Q}_p^n\mathbf{c}_p^{n+1}\\ &= (\mathbf{Q}_p^{n-T}\mathbf{Q}_p^{nT})\mathbf{M}_p^n\mathbf{Q}_p^n\mathbf{c}_p^{n+1}\\ &= (\mathbf{Q}_p^{n-T}\mathbf{Q}_p^{nT})\mathbf{M}_p^n\mathbf{Q}_p^n(\mathbf{Q}_p^{nT}\mathbf{M}_p^n\mathbf{Q}_p^n)^{-1}\mathbf{Q}_p^{nT}\mathbf{M}_p^n\mathcal{V}_p^n\\ &= (\mathbf{Q}_p^n)^{-T}(\mathbf{Q}_p^{nT}\mathbf{M}_p^n\mathbf{Q}_p^n)(\mathbf{Q}_p^{nT}\mathbf{M}_p^n\mathbf{Q}_p^n)^{-1}\mathbf{Q}_p^{nT}\mathbf{M}_p^n\mathcal{V}_p^n\\ &= \mathbf{M}_p^n\mathcal{V}_p^n\end{aligned}$$

A.4 Linear and Angular Momentum Conservation

The ith component of the local linear momentum associated with velocity $\mathcal{U}$ is $\mathbf{C}_i^T(\mathbf{Q}_p^n)^T\mathbf{M}_p^n\mathcal{U}$ and the j^{th} component of the angular velocity is $\mathbf{C}_{j+d}^T(\mathbf{Q}_p^n)^T\mathbf{M}_p^n\mathcal{U}$ where $C_{ki} = \delta_{ki}$ for $1 \leq k \leq d$ and

$$C_{k(j+d)} = \begin{cases} 1, \ k = 4 \\ -1, \ k = 5 \end{cases}$$

when $d = 2$ and

$$C_{k4} = \begin{cases} 1, \ k = 5 \\ -1, \ k = 7 \end{cases}$$

$$C_{k5} = \begin{cases} 1, \ k = 6 \\ -1, \ k = 10 \end{cases}$$

$$C_{k6} = \begin{cases} 1, \ k = 9 \\ -1, \ k = 11 \end{cases}$$

Thus local linear and angular momentum conservation (which implies global) follows from

$$\mathbf{C}_i^T(\mathbf{Q}_p^n)^T\mathbf{M}_p^n\mathcal{V}_p^{n+1} = \mathbf{C}_i^T(\mathbf{Q}_p^n)^T\mathbf{M}_p^n\mathbf{Q}_p^n\mathbf{c}_p^{n+1}$$

A.5 Grid Momentum Update

In the case of the incompressible Euler, we used a MAC grid discretization of the pressure projection to update the fluid velocity. We first compute an intermediate velocity field on the grid

$$\mathbf{v}_{\mathbf{i}}^{*} = \mathbf{v}_{\mathbf{i}}^{n} + \Delta t \mathbf{g},$$

where $\mathbf{g}$ is the gravitational acceleration. Then we set up a Poisson system for pressure p

$$\nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{v}_{\mathbf{i}}^{*}$$

to project grid velocities to be divergence free. Note that velocity is discretized at cell faces and pressure is discretized at cell centers. During the solve we enforce Dirichlet $p = 0$ boundary condition at free surface cells and zero Neumann boundary condition at collision object cells. Then we compute the updated velocity with

$$\hat{\mathbf{v}}_{\mathbf{i}}^{n+1} = \mathbf{v}_{\mathbf{i}}^{*} + \frac{\Delta t}{\rho} \nabla p.$$

In the case of elastoplastic solids and MPM the updates is from the elastic force.

$$\hat{\mathbf{v}}_{\mathbf{i}}^{n+1} = \mathbf{v}_{\mathbf{i}}^{n} + \frac{\Delta t}{m_{\mathbf{i}}^{n}} (\mathbf{f} + \mathbf{g})$$

where $\mathbf{f}$ is the elastic force. If explicit time integration is adopted, $\mathbf{f}_{\mathbf{i}}^{n}$ is given explicitly as

$$\mathbf{f}_{\mathbf{i}}^{n} = -\sum_{p} V_p^n \boldsymbol{\sigma}_p^n \nabla w_{\mathbf{i}p}^n,$$

where $V_p^n = V_p^0 J_p^n$ is the current volume of particle p, V_p^0 is its original volume, $J_p^n = \det(\mathbf{F}_p^n)$ and $\mathbf{F}_p^n$ is the current deformation gradient. Cauchy stress $\boldsymbol{\sigma}_p^n$ can be computed from $\mathbf{F}_p^n$ using any elastic constitutive model. For hyperelasticity with energy density function $\Psi(\mathbf{F})$, $\boldsymbol{\sigma}_p^n$ is given by

$$\boldsymbol{\sigma}_p^n = \frac{1}{J_p^n} \frac{\partial \Psi}{\partial \mathbf{F}} (\mathbf{F}_p^n) \mathbf{F}_p^{nT}.$$

For implicit time integration such as Backward Euler, we need to differentiate the force with respect to imaginarily deformed grid node positions $\hat{\mathbf{x}}_{\mathbf{i}} = \mathbf{x}_{\mathbf{i}}^n + \Delta t \hat{\mathbf{v}}_{\mathbf{i}}$. The force differential on an arbirary increment $\delta \mathbf{u}$ is given by

$$\delta f_{i\alpha} = \frac{\partial f_{i\alpha}}{\partial x_{j\lambda}} \delta u_{j\lambda} = -\sum_p V_p^0 \frac{\partial F_{\beta\zeta}^E}{\partial x_{i\alpha}} \frac{\partial^2 \psi}{\partial F_{\beta\zeta}^E \partial F_{\omega\sigma}^E} \frac{\partial F_{\omega\sigma}^E}{\partial x_{j\lambda}} \delta u_{j\lambda}.$$

Based on that we solve

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

using Newton's method.

A.6 Code

In this section we present the Mathematica code generating the bases and corresponding formula.

A.6.1 Linear interpolation in 2D

```
 (* 2D linear *)
 ClearAll[ “Global`*”]
 (* Assume x is in [0, h],
 N[x, 1] is the weight of node at xi = 0,
 N[x, 2] is the weight of node at h.
 *)
 N1[ x_, i_] = Piecewise[{{1 − x, i == 1}, {x, i == 2}}];
 NN[ x_, y_, ii_, jj_] = N1[x/h, ii] ∗ N1[y/h, jj];
 (* xi - xp *)
r = ConstantArray[0, {4, 2}];
 Do[{id = 2 ∗ ( i1 − 1) + j1;
 nodex = ( i1 − 1) ∗ h;
 nodey = ( j1 − 1) ∗ h;
r[[id]][[1]] = (nodex −  particlex)/h;
r[[id]][[2]] = (nodey −  particley)/h; }, { i1, 1, 2}, { j1, 1, 2}];
 (* mass *)
M =  ConstantArray[0, {4, 4}];
 Do[{id = 2 ∗ ( i1 − 1) +  j1;
 weight =  NN[ particlex,  particley,  i1,  j1];
M[[id]][[id]] =  mass ∗  weight; }, { i1, 1, 2}, { j1, 1, 2}];
 (* basis *)
 (* row index is node index
  col index is basis index
 *)
 Do[{id = 2 ∗ ( i1 − 1) +  j1;
 weight =  NN[ particlex,  particley,  i1,  j1];
M[[id]][[id]] =  mass ∗  weight; }, { i1, 1, 2}, { j1, 1, 2}];
 B1 = ConstantArray[0, {4, 4}];
 Do[{id = ( i1 − 1) ∗ 2 +  j1;
 B1[[idr]][[id]]  = r[[idr]][[1]]^( i1 − 1) ∗ r[[idr]][[2]]^( j1 − 1)}, { i1, 1, 2}, { j1, 1, 2}, {idr, 1, 4}];
 Diagonal[M] // MatrixForm;
 (* Verify the diagonal structure *)
 MatrixForm[ Transpose[ B1].M. B1] // Simplify
```

A.6.2 Linear interpolation in 3D

```
 (* 3D linear *)
 ClearAll[ “Global`*”]
 (* Assume x is in[0, h],
 N[x, 1] is the weight of node at xi = 0,
 N[x, 2] is the weight of node ath.
 *)
 N1[ x_, i_] = Piecewise[{{1 − x, i == 1}, {x, i == 2}}];
 NN[ x_, y_, z_, ii_, jj_, kk_] =  N1[x/h, ii] ∗  N1[y/h, jj] ∗  N1[z/h, kk];
 (* xi - xp *)
r = ConstantArray[0, {8, 3}];
 Do[{id = 4 ∗ ( i1 − 1) + 2 ∗ ( j1 − 1) +  k1;
 nodex = ( i1 − 1) ∗ h;
 nodey = ( j1 − 1) ∗ h;
 nodez = ( k1 − 1) ∗ h;
r[[id]][[1]] = ( nodex −  particlex)/h;
r[[id]][[2]] = ( nodey −  particley)/h;
r[[id]][[3]] = ( nodez −  particlez)/h; }, { i1, 1, 2}, { j1, 1, 2}, { k1, 1, 2}];
 (* mass *)
M = ConstantArray[0, {8, 8}];
 Do[{id = 4 ∗ ( i1 − 1) + 2 ∗ ( j1 − 1) +  k1;
 weight =  NN[ particlex,  particley,  particlez,  i1,  j1,  k1];
M[[id]][[id]] =  mass ∗  weight; }, { i1, 1, 2}, { j1, 1, 2}, { k1, 1, 2}];
 (* basis *)
 (* row index is node index
  col index is basis index
 *)
 B1 =  ConstantArray[0, {4 ∗ 2, 8}];
 Do[{id = ( i1 − 1) ∗ 4 + ( j1 − 1) ∗ 2 +  k1;
 B1[[idr]][[id]]  = r[[idr]][[1]]^( i1 − 1) ∗ r[[idr]][[2]]^( j1 − 1) ∗ r[[idr]][[3]]^( k1 − 1); },
      {i1, 1, 2}, { j1, 1, 2}, { k1, 1, 2}, {idr, 1, 8}];
 Diagonal[M] // MatrixForm;
 (* Verify the diagonal structure *)
 MatrixForm[ Transpose[ B1].M. B1] // Simplify
```

A.6.3 Quadratic interpolation in 2D

```
(* 2D quadratic *)
ClearAll[“Global`*”]
(* Assumexis in[−0.5h, 0.5h], we use quadratic interpolation.
N2[x, 1]is the weight of node at xi = −h,
 N2[x, 2]is the weight of node at xi = 0,
 N2[x, 3]is the weight of node at xi = h,
*)
N2[x_, i_] = Piecewise[{{1/2 ∗ (1/2 − x)^2, i == 1},
{3/4 − x^2, i == 2}, {1/2 ∗ (x + 1/2)^2, i == 3}}];
NN[x_, y_, ii_, jj_] = N2[x/h, ii] ∗ N2[y/h, jj];
(* xi - xp *)
r = ConstantArray[0, {9, 2}];
Do[{id = 3 ∗ (i1 − 1) + j1;
nodex = (i1 − 2) ∗ h;
nodey = (j1 − 2) ∗ h;
r[[id]][[1]] = (nodex − x);
r[[id]][[2]] = (nodey − y); }, {i1, 1, 3}, {j1, 1, 3}];
(* mass *)
M = ConstantArray[0, {9, 9}];
Do[{id = 3 ∗ (i1 − 1) + j1;
weight = NN[x, y, i1, j1];
M[[id]][[id]] = mass ∗ weight; }, {i1, 1, 3}, {j1, 1, 3}];
(* basis *)
(* row index is node index
 col index is basis index
*)
B = ConstantArray[0, {9, 9}];
Do[{id = (j1 − 1) ∗ 3 + i1;
B[[idr]][[id]]  = r[[idr]][[1]]^(i1 − 1) ∗ r[[idr]][[2]]^(j1 − 1);
}, {i1, 1, 3}, {j1, 1, 3}, {idr, 1, 9}];
(* Rearrange the basis so that the corresponding polynomials are in the order of
 1, x, y, xy, x^2, y^2, x^2 ∗ y, x ∗ y^2, x^2 ∗ y^2.
*)
BS = B[[All, {1, 2, 4, 5, 3, 7, 6, 8, 9}]];
(* The first four basis vectors are already othogonal.
Use Gram-Schmidt and we get the basis vectors corresponding to x^2 and y^2.
*)
BS[[All, 5]] = BS[[All, 5]] − h^2/4 − x(h^2 − 4x^2)/h^2 ∗ BS[[All, 2]];
BS[[All, 6]] = BS[[All, 6]] − h^2/4 − y(h^2 − 4y^2)/h^2 ∗ BS[[All, 3]];
(* The rest are simply products of the previous ones *)
BS[[All, 7]] = BS[[All, 5]]BS[[All, 3]];
BS[[All, 8]] = BS[[All, 6]]BS[[All, 2]];
BS[[All, 9]] = BS[[All, 5]] BS[[All, 6]];
(* Verify the diagonal structure *)
BTMB = Transpose[BS].M.BS//Simplify;
(* Get the awesome formula to put in your code!*)
MB = M.BS//Simplify;
MB//MatrixForm;
Inverse[BTMB].Transpose[BS].M//MatrixForm//Simplify;
```

A.6.4 Quadratic interpolation in 3D

```
 (* 3D quadratic *)
 ClearAll[ “Global*”]
 (* Assumex is in[−0.5h, 0.5h], we use quadratic interpolation.
 N2[x, 1] is the weight of node at xi = −h,
  N2[x, 2] is the weight of node at xi = 0,
  N2[x, 3] is the weight of node at xi = h,
 *)
 N2[ x_, i_] =  Piecewise[{{1/2 * (1/2 - x)^2, i == 1},
{3/4 - x^2, i == 2}, {1/2 * (x + 1/2)^2, i == 3}}];
 NN[ x_, y_, z_, ii_, jj_, kk_] =  N2[x/h,  ii] *  N2[y/h,  jj] *  N2[z/h,  kk];
 (* xi - xp *)
r =  ConstantArray[0, {27, 3}];
 Do[{id = 9 * ( i1 - 1) + 3 * ( j1 - 1) +  k1;
 nodex = ( i1 - 2) * h;
 nodey = ( j1 - 2) * h;
 nodez = ( k1 - 2) * h;
r[[id]][[1]] = ( nodex - x);
r[[id]][[2]] = ( nodey - y);
r[[id]][[3]] = ( nodez - z); }, { i1, 1, 3}, { j1, 1, 3}, { k1, 1, 3}];
 (* mass *)
M =  ConstantArray[0, {27, 27}];
 Do[{id = 9 * ( i1 - 1) + 3 * ( j1 - 1) +  k1;
 weight =  NN[x, y, z,  i1,  j1,  k1];
M[[id]][[id]] =  mass *  weight; }, { i1, 1, 3}, { j1, 1, 3}, { k1, 1, 3}];
 (* basis *)
 (* row index is node index
  col index is basis index
 *)
B =  ConstantArray[0, {27, 27}];
 Do[{id = ( i1 - 1) * 4 + ( j1 - 1) * 2 +  k1;
B[[idr]][[id]]  = r[[idr]][[1]]^( i1 - 1) * r[[idr]][[2]]^( j1 - 1) * r[[idr]][[3]]^( k1 - 1);
}, { i1, 1, 2}, { j1, 1, 2}, { k1, 1, 2}, {idr, 1, 27}];
 (* The basisvectors  corresponding to1, x, y, z are already orthogonal.
 Use gram -  schmidt and we get the basis corresponding to x^2, y^2,  andz^2.
 *)
```

```
B[[All, 9]] = B[[All, 5]]B[[All, 5]] − h^2/4 − x(h^2 − 4x^2)/h^2 ∗ B[[All, 5]];
B[[All, 10]] = B[[All, 3]]B[[All, 3]] − h^2/4 − y(h^2 − 4y^2)/h^2 ∗ B[[All, 3]];
B[[All, 11]] = B[[All, 2]]B[[All, 2]] − h^2/4 − z(h^2 − 4z^2)/h^2 ∗ B[[All, 2]];

 (* The rest are simply products of the previous ones *)
B[[All, 12]] = B[[All, 2]]B[[All, 9]];
B[[All, 13]] = B[[All, 3]]B[[All, 9]];
B[[All, 14]] = B[[All, 2]]B[[All, 3]]B[[All, 9]];

B[[All, 15]] = B[[All, 2]]B[[All, 10]];
B[[All, 16]] = B[[All, 5]]B[[All, 10]];
B[[All, 17]] = B[[All, 2]]B[[All, 5]]B[[All, 10]];

B[[All, 18]] = B[[All, 3]]B[[All, 11]];
B[[All, 19]] = B[[All, 5]]B[[All, 11]];
B[[All, 20]] = B[[All, 3]]B[[All, 5]]B[[All, 11]];

B[[All, 21]] = B[[All, 9]]B[[All, 10]];
B[[All, 22]] = B[[All, 10]]B[[All, 11]];
B[[All, 23]] = B[[All, 9]]B[[All, 11]];

B[[All, 24]] = B[[All, 21]]B[[All, 2]];
B[[All, 25]] = B[[All, 22]]B[[All, 5]];
B[[All, 26]] = B[[All, 23]]B[[All, 3]];

B[[All, 27]] = B[[All, 9]]B[[All, 10]]B[[All, 11]];

 (*  Rearrange the basis so that the corresponding polynomials are in the order of
1, x, y, z,  xy,  xz,  yz,  xyz, x^2, y^2, z^2, x^2 ∗ y^2, x^2 ∗ z^2, y^2 ∗ z^2, x^2 ∗ y^2 ∗ z^2,
x^2 * y,  x^2 * z,  x^2 *  yz,  y^2 * x,y^2 * z,y^2 *  xz,z^2 * x,z^2 * y,z^2 *  xy,
x^2 * y^2 * z,x^2 * z^2 * y,y^2 * z^2 * x
*)
BS = B[[All, {1, 5, 3, 2, 7, 6, 4, 8, 9, 10, 11, 21, 23, 22, 27, 13, 12, 14, 16,
15, 17, 19, 18, 20, 24, 26, 25}]];

(* Verify the diagonal structure *)
BTMB =  Transpose[ BS].M. BS // Simplify;
BTMB // MatrixForm
(* Get the awesome formula to put in your code! *)
MB = M. BS // Simplify;
MB // MatrixForm;
BTMBinvBTM =  Inverse[ BTMB]. Transpose[ BS].M;
Transpose[ BTMBinvBTM] // MatrixForm // Simplify;
```

APPENDIX B

Numerical Simulation of Thin Shell With Frictional Contact

B.1 FEM Force Computation

We compute forces on the control points $\mathbf{x}_p$ by

$$
\begin{aligned}
\mathbf{f}_p^{KL} &= -\frac{\partial \Psi^S(\mathbf{F}^{KL,Etr}(\mathbf{x}^{KL}))}{\partial \mathbf{x}_p^{KL}} \\
&= -\sum_q V_q^0 \frac{\partial \psi(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}_q))}{\partial \mathbf{x}_p^{KL}} \\
&= -\sum_q V_q^0 \frac{\partial \psi}{\partial \mathbf{F}^{KL}}(\mathbf{F}_q^{KL,E\mathrm{tr}}(\mathbf{x}_q)) : \frac{\partial \mathbf{F}_q^{KL,E\mathrm{tr}}}{\partial \mathbf{x}_p^{KL}}(\mathbf{x}_q),
\end{aligned}
$$

where $\mathbf{x}_q$'s are positions of the quadrature points. We give expressions for each $\frac{\partial \mathbf{F}_q^{KL}}{\partial x_{p_k}^{KL}}(\mathbf{x}_q)$ with fixed p, q and k, where k represents the x, y, or z direction. For simplicity of notation, we omit the subscripts p, q and superscript KL for now.

Recall from the paper that we have

$$
\mathbf{F} = \sum_{i=1}^{3} \mathbf{g}_i \otimes \bar{\mathbf{g}}^i, \text{ with } \mathbf{g}_\alpha = \mathbf{a}_\alpha + \xi_3 \mathbf{a}_{3,\alpha},\ \mathbf{g}_3 = \mathbf{a}_3,
$$

where

$$\begin{aligned}
\mathbf{a}_\alpha &= \sum_j \mathbf{x}_j \frac{\partial N_j^{SD}}{\partial \xi_\alpha}(\xi_1, \xi_2), \quad \alpha = 1, 2 \\
\mathbf{a}_3 &= \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|} \\
\mathbf{a}_{3,\alpha} &= (\mathbf{I} - \mathbf{a}_3 \otimes \mathbf{a}_3)\frac{\mathbf{a}_{1,\alpha} \times \mathbf{a}_2 + \mathbf{a}_1 \times \mathbf{a}_{2,\alpha}}{|\mathbf{a}_1 \times \mathbf{a}_2|} \\
&= \tilde{\mathbf{a}} - \mathbf{a}_3(\mathbf{a}_3 \cdot \tilde{\mathbf{a}})
\end{aligned}$$

in which we define $\tilde{\mathbf{a}}$ to be

$$\tilde{\mathbf{a}} = \frac{\mathbf{a}_{1,\alpha} \times \mathbf{a}_2 + \mathbf{a}_1 \times \mathbf{a}_{2,\alpha}}{|\mathbf{a}_1 \times \mathbf{a}_2|}.$$

Now we compute $\frac{\partial \mathbf{F}}{\partial x_k}$.

$$\frac{\partial \mathbf{F}}{\partial x_k} = \sum_{i=1}^{3} \frac{\partial \mathbf{g}_i}{\partial x_k} \otimes \bar{\mathbf{g}}^i,$$

and

$$\begin{aligned}
\frac{\partial \mathbf{g}_\alpha}{\partial x_k} &= \frac{\partial \mathbf{a}_\alpha}{\partial x_k} + \xi_3 \frac{\partial \mathbf{a}_{3,\alpha}}{\partial x_k} \\
\frac{\partial \mathbf{g}_3}{\partial x_k} &= \frac{\partial \mathbf{a}_3}{\partial x_k}
\end{aligned}$$

where

$$\frac{\partial \mathbf{a}_\alpha}{\partial x_k} = \frac{\partial N_k^{SD}(\xi_1, \xi_2)}{\partial \xi_\alpha}\mathbf{e}_k \text{ (summation convention does not apply here)} \tag{B.1}$$

$$\frac{\partial \mathbf{a}_3}{\partial x_k} = \frac{\frac{\partial \mathbf{a}_1}{\partial x_k} \times \mathbf{a}_2 + \mathbf{a}_1 \times \frac{\partial \mathbf{a}_2}{\partial x_k} - \frac{|\mathbf{a}_1 \times \mathbf{a}_2|}{\partial x_k}\mathbf{a}_3}{|\mathbf{a}_1 \times \mathbf{a}_2|},$$

and

$$\frac{|\mathbf{a}_1 \times \mathbf{a}_2|}{\partial x_k} = \mathbf{a}_3 \cdot (\frac{\partial \mathbf{a}_1}{\partial x_k} \times \mathbf{a}_2 + \mathbf{a}_1 \times \frac{\partial \mathbf{a}_2}{\partial x_k})$$

Finally,

$$\frac{\partial \mathbf{a}_{3,\alpha}}{\partial x_k} = \frac{\partial \tilde{\mathbf{a}}}{\partial x_k} - \mathbf{a}_3(\frac{\partial \mathbf{a}_3}{\partial x_k} \cdot \tilde{\mathbf{a}} + \mathbf{a}_3 \cdot \frac{\partial \tilde{\mathbf{a}}}{\partial x_k}) - \frac{\partial \mathbf{a}_3}{\partial x_k}(\mathbf{a}_3 \cdot \tilde{\mathbf{a}}),$$

where

$$\frac{\partial \tilde{\mathbf{a}}}{\partial x_k} = \frac{\frac{\mathbf{a}_{1,\alpha}}{\partial x_k} \times \mathbf{a}_2 + \mathbf{a}_{1,\alpha} \times \frac{\partial \mathbf{a}_2}{\partial x_k} + \frac{\partial \mathbf{a}_1}{\partial x_k} \times \mathbf{a}_{2,\alpha} + \mathbf{a}_1 \times \frac{\partial \mathbf{a}_{2,\alpha}}{\partial x_k}}{|\mathbf{a}_1 \times \mathbf{a}_2|} - \frac{\mathbf{a}_{1,\alpha} \times \mathbf{a}_2 + \mathbf{a}_1 \times \mathbf{a}_{2,\alpha}}{|\mathbf{a}_1 \times \mathbf{a}_2|^2} \frac{\partial |\mathbf{a}_1 \times \mathbf{a}_2|}{\partial x_k},$$

in which

$$\frac{\partial \mathbf{a}_{\alpha,\beta}}{\partial x_k} = \frac{N_k^{SD}(\xi_1, \xi_2)}{\partial \xi_\beta \partial \xi_\alpha} \mathbf{e}_k \text{ (summation convention does not apply here).}$$

B.2 Grid force Computation

The force on the MPM grid $f_{\mathbf{i}}^{iii}(\mathbf{x}^*)$ computes as follows:

$$\begin{aligned}
\mathbf{f}_{\mathbf{i}}^{(iii)}(\mathbf{x}^*) &= \sum_{p \in \mathcal{I}^{(iii)}} \frac{\partial \chi(\mathbf{a}_{p\alpha} \otimes \bar{\mathbf{a}}_{p\alpha} + \mathbf{a}_{p3}^E \otimes \bar{\mathbf{a}}_{p3})}{\partial \mathbf{x}_{\mathbf{i}}} V_p^0 \\
&= \sum_{p \in \mathcal{I}^{(iii)}} \frac{\partial \chi(\mathbf{a}_{p\alpha} \otimes \bar{\mathbf{a}}_{p\alpha} + \mathbf{a}_{p3}^E \otimes \bar{\mathbf{a}}_{p3})}{\partial \mathbf{F}^E} : \frac{\partial \left(\mathbf{a}_{p\alpha} \otimes \bar{\mathbf{a}}_{p\alpha} + \mathbf{a}_{p3}^E \otimes \bar{\mathbf{a}}_{p3}\right)}{\partial \mathbf{a}_{p\beta}} : \frac{\partial \mathbf{a}_{p\beta}}{\mathbf{x}_{\mathbf{i}}} V_p^0 \\
&+ \sum_{p \in \mathcal{I}^{(iii)}} \frac{\partial \chi(\mathbf{a}_{p\alpha} \otimes \bar{\mathbf{a}}_{p\alpha} + \mathbf{a}_{p3}^E \otimes \bar{\mathbf{a}}_{p3})}{\partial \mathbf{F}^E} : \frac{\partial \left(\mathbf{a}_{p\alpha} \otimes \bar{\mathbf{a}}_{p\alpha} + \mathbf{a}_{p3}^E \otimes \bar{\mathbf{a}}_{p3}\right)}{\partial \mathbf{a}_{p3}^E} : \frac{\partial \mathbf{a}_{p3}^E}{\mathbf{x}_{\mathbf{i}}} V_p^0.
\end{aligned}$$

Then, omitting the subscript p, we compute each term in the contraction:

$$\begin{aligned}
\frac{\partial \chi(\mathbf{a}_\alpha \otimes \bar{\mathbf{a}}_\alpha + \mathbf{a}_3^E \otimes \bar{\mathbf{a}}_3)}{\partial \mathbf{F}^E} &= \boldsymbol{\tau}^S \left(\mathbf{a}_\alpha \otimes \bar{\mathbf{a}}_\alpha + \mathbf{a}_3^E \otimes \bar{\mathbf{a}}_3)\right)^{-T} \\
&= \boldsymbol{\tau}^S \left(\tilde{\mathbf{a}}^\alpha \otimes \bar{\mathbf{a}}^\alpha + \tilde{\mathbf{a}}^3 \otimes \bar{\mathbf{a}}_3)\right)
\end{aligned}$$

where $\boldsymbol{\tau}^S$ is the Kirchhoff stress and $\tilde{\mathbf{a}}^\alpha$ and $\tilde{\mathbf{a}}^3$ are the contravariant counterparts of $\mathbf{a}_\alpha$ and $\mathbf{a}_3^E$ respectively.

And using index notation, we see that

$$\begin{aligned}\frac{\partial\left(\mathbf{a}_\alpha\otimes\bar{\mathbf{a}}_\alpha+\mathbf{a}_3^E\otimes\bar{\mathbf{a}}_3\right)}{\partial\mathbf{a}_\beta}&=\frac{\partial a_{\alpha_i}\bar{a}_{\alpha_j}}{\partial a_{\beta_k}}\\&=\delta_{\alpha\beta}\delta_{ik}\bar{a}_{\alpha_j}\\&=\delta_{ik}\bar{a}_{\beta_j}\end{aligned}$$

Similarly,

$$\frac{\partial\left(\mathbf{a}_\alpha\otimes\bar{\mathbf{a}}_\alpha+\mathbf{a}_3^E\otimes\bar{\mathbf{a}}_3\right)}{\partial\mathbf{a}_3^E}=\delta_{ik}\bar{a}_{3_j}$$

Hence, contracting the first two terms in the summation, each term in the summation becomes

$$\begin{aligned}&\boldsymbol{\tau}^S\left(\tilde{\mathbf{a}}^\alpha\otimes\bar{\mathbf{a}}^\alpha+\tilde{\mathbf{a}}_3\otimes\bar{\mathbf{a}}_3\right)\bar{\mathbf{a}}_\beta:\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_\mathbf{i}}+\boldsymbol{\tau}^S\left(\tilde{\mathbf{a}}^\alpha\otimes\bar{\mathbf{a}}^\alpha+\tilde{\mathbf{a}}_3\otimes\bar{\mathbf{a}}_3\right)\bar{\mathbf{a}}_3:\frac{\partial\mathbf{a}_3^E}{\partial\mathbf{x}_\mathbf{i}}\\&=\boldsymbol{\tau}^S\tilde{\mathbf{a}}^\beta:\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_\mathbf{i}}+\boldsymbol{\tau}^S\tilde{\mathbf{a}}^3:\frac{\partial\mathbf{a}_3^E}{\partial\mathbf{x}_\mathbf{i}}\end{aligned}$$

Note that

$$\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_\mathbf{i}}=\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_p}\frac{\partial\mathbf{x}_p}{\partial\mathbf{x}_\mathbf{i}}=\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_p}w_{\mathbf{i}p}^n,$$

and the expression for $\frac{\partial\mathbf{a}_\beta}{\partial\mathbf{x}_p}$ is given equation (B.1).

Ignoring further plastic flow, we have

$$\mathbf{a}_3^E(\mathbf{x}^*)=\left(\sum_\mathbf{j}\mathbf{x}_\mathbf{j}^*\otimes\nabla w_{\mathbf{j}p}^n\right)\mathbf{a}_3^{E,n},$$

and thus,

$$\frac{\partial \mathbf{a}_3^E}{\partial \mathbf{x_i}} = \nabla w_{\mathbf{i}p}^n \mathbf{a}_3^{E,n}$$

Therefore, we arrive at the final expression for the force of type (iii):

$$\mathbf{f}_{\mathbf{i}}^{(iii)}(\mathbf{x}^*) = \sum_{p \in \mathcal{I}^{(iii)}} \boldsymbol{\tau}_p^S \tilde{\mathbf{a}}_p^\beta : \frac{\partial \mathbf{a}_{p\beta}}{\partial \mathbf{x}_p} w_{\mathbf{i}p}^n + \boldsymbol{\tau}_p^S \tilde{\mathbf{a}}_p^3 : \nabla w_{\mathbf{i}p}^n \mathbf{a}_{p3}^{E,n}$$

B.3 Laminate Stress

In this section we derive the expression for

$$\boldsymbol{\tau}^{KL} = \tau_{\alpha\beta} \mathbf{q}_\alpha^{KL,E} \otimes \mathbf{q}_\beta^{KL,E}, \quad \tau_{\alpha\beta}^{KL} = 2\mu\epsilon_{\alpha\beta}^L + \lambda\epsilon_{\gamma\gamma}^L \delta_{\alpha\beta}. \tag{B.2}$$

First notice that we may replace the right Hencky strain with left Hencky strain in the definition of energy because of the isotropic nature of the energy function. We now give the drivation of Equation (B.2) with index free notation assuming all variables are in 2D.

$$\begin{aligned}\psi(\mathbf{F}) &= \psi(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T) \\ \mathbf{P}(\mathbf{F}) &= \mathbf{P}(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T) = \mathbf{U}\mathbf{P}(\boldsymbol{\Sigma})\mathbf{V}^T\end{aligned}$$

because the energy is isotropic.

Hence,

$$\begin{aligned}\mathbf{P}(\mathbf{F}) &= \mathbf{U}\mathbf{P}(\boldsymbol{\Sigma})\mathbf{V}^T \\ &= \mathbf{U}\frac{\partial \psi}{\partial \boldsymbol{\Sigma}}\mathbf{V}^T \\ &= \mathbf{U}\left(2\mu \log(\boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-1} + \lambda \mathrm{tr}(\log \boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-1}\right)\mathbf{V}^T.\end{aligned}$$

Therefore,

$$\begin{aligned}\boldsymbol{\tau}^{KL} &= \left(\mathbf{U}\left(2\mu\log(\boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-1} + \lambda\mathrm{tr}(\log\boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-1}\right)\mathbf{V}^T\right)\mathbf{F}^T \\ &= \mathbf{U}\left(2\mu\log(\boldsymbol{\Sigma}) + \lambda\mathrm{tr}(\log\boldsymbol{\Sigma})\right)\mathbf{U}^T \\ &= 2\mu\boldsymbol{\epsilon}^L + \lambda\mathrm{tr}(\boldsymbol{\epsilon}^L)\end{aligned}$$

B.4 QR and Elastic Potential

We can use QR orthogonalization of deformed material directions to define

$$\mathbf{q}_i r_{ij} = \mathbf{F}\bar{\mathbf{a}}_j,\ \mathbf{F} = r_{ij}\mathbf{q}_i \otimes \bar{\mathbf{a}}_j,\ r_{ij} = 0 \text{ for } i > j. \tag{B.3}$$

B.4.1 Change of basis tensor

Define the change of basis tensor

$$\mathbf{Q} = Q_{ij}\bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j \tag{B.4}$$

with $Q_{ij} = \mathbf{q}_j \cdot \bar{\mathbf{a}}_i$. With this convention we see that $\mathbf{Q}\bar{\mathbf{a}}_i = \mathbf{q}_i$ and $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$. Furthermore, defining

$$\mathbf{R} = r_{ij}\bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j$$

we have $\mathbf{F} = \mathbf{QR}$.

B.4.2 Differentials

The QR differential satisfies

$$\mathbf{q}_k \cdot \delta\mathbf{q}_i r_{ij} + \delta r_{kj} = \mathbf{q}_k \cdot (\delta\mathbf{F}\bar{\mathbf{a}}_j),\ \delta\mathbf{F} = \delta r_{ij}\mathbf{q}_i \otimes \bar{\mathbf{a}}_j + r_{ij}\delta\mathbf{q}_i \otimes \bar{\mathbf{a}}_j \tag{B.5}$$

where $\mathbf{q}_k \cdot \delta\mathbf{q}_i = -\mathbf{q}_i \cdot \delta\mathbf{q}_k$ from orthogonality of the $\mathbf{q}_i$. And

$$\delta\mathbf{F} = \delta\mathbf{Q}\mathbf{R} + \mathbf{Q}\delta\mathbf{R} \tag{B.6}$$

where $\delta\mathbf{Q}^T\mathbf{Q} = -\mathbf{Q}^T\delta\mathbf{Q}$ from $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$. Furthermore,

$$\delta\mathbf{Q} = \delta Q_{ij}\bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j,\ \delta Q_{ij} = \delta\mathbf{q}_j \cdot \bar{\mathbf{a}}_i,\ \delta\mathbf{q}_i = \delta\mathbf{Q}\bar{\mathbf{a}}_i \tag{B.7}$$

$$\delta\mathbf{R} = \delta r_{ij}\bar{\mathbf{a}}_i \otimes \bar{\mathbf{a}}_j \tag{B.8}$$

and the $\delta r_{ij} = 0$ for $i > j$.

B.5 Elastic Potential and Stresses

Define the hyperelastic potential as

$$\psi(\mathbf{F}) = \hat{\psi}([\mathbf{R}]) \tag{B.9}$$

where

$$[\mathbf{R}] = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ & r_{22} & r_{23} \\ & & r_{33} \end{pmatrix}. \tag{B.10}$$

The differential satisfies

$$\delta\psi(\mathbf{F}) = \frac{\partial\psi}{\partial\mathbf{F}}(\mathbf{F}) : \delta\mathbf{F} = \mathbf{P} : \delta\mathbf{F} = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])\delta r_{ij} \tag{B.11}$$

where $\mathbf{P} = \frac{\partial\psi}{\partial\mathbf{F}}(\mathbf{F})$. Therefore

$$\delta r_{ij}\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j) + r_{ij}\delta\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j) = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])\delta r_{ij}. \tag{B.12}$$

Similarly,

$$\mathbf{P} : \delta\mathbf{F} = \mathbf{P} : (\delta\mathbf{Q}\mathbf{R}) + \mathbf{P} : (\mathbf{Q}\delta\mathbf{R}) = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])\delta r_{ij} \tag{B.13}$$

Choosing $\delta\mathbf{F} = \delta r_{ij}\mathbf{q}_i \otimes \bar{\mathbf{a}}_j$ (i.e. $\delta\mathbf{q}_i = \mathbf{0}$), we can conclude that

$$\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j)\, \delta r_{ij} = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])\delta r_{ij} \tag{B.14}$$

for arbitrary δr_{ij} with $i \leq j$. Therefore the $\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j) = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])$ for $i \leq j$. Similarly,

$$\mathbf{P} : (\mathbf{Q}\delta\mathbf{R}) = (\mathbf{Q}^T\mathbf{P}) : \delta\mathbf{R} = \delta r_{ij}\bar{\mathbf{a}}_i \cdot \left(\mathbf{Q}^T\mathbf{P}\bar{\mathbf{a}}_j\right) = \delta r_{ij}\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j) = \frac{\partial\hat{\psi}}{\partial r_{ij}}([\mathbf{R}])\delta r_{ij}. \tag{B.15}$$

Choosing $\delta\mathbf{F} = r_{ij}\delta\mathbf{q}_i \otimes \bar{\mathbf{a}}_j$ (i.e. $\delta r_{ij} = 0$), we can conclude that

$$0 = r_{ij}\delta\mathbf{q}_i \cdot (\mathbf{P}\bar{\mathbf{a}}_j)\,. \tag{B.16}$$

Similarly,

$$0 = \mathbf{P} : (\delta\mathbf{Q}\mathbf{R}) = \left(\mathbf{P}\mathbf{R}^T\right) : \delta\mathbf{Q} = \left(\mathbf{P}\mathbf{R}^T\right) : \left(\delta\mathbf{Q}\mathbf{Q}^T\mathbf{Q}\right) = \left(\mathbf{P}\mathbf{F}^T\right) : \left(\delta\mathbf{Q}\mathbf{Q}^T\right) \tag{B.17}$$

In other words, the Kirchhoff stress $\boldsymbol{\tau} = \mathbf{P}\mathbf{F}^T$ is symmetric since $\delta\mathbf{Q}\mathbf{Q}^T$ is arbitrary skew. Furthermore,

$$\mathbf{P} = P_{ij}\mathbf{q}_i \otimes \bar{\mathbf{a}}_j,\ \boldsymbol{\tau} = P_{ij}r_{kj}\mathbf{q}_i \otimes \mathbf{q}_k = \tau_{ik}\mathbf{q}_i \otimes \mathbf{q}_k \tag{B.18}$$

and we know $P_{ij} = \frac{\partial \hat{\psi}}{\partial r_{ij}}$ for $i \leq j$ from Equation B.14. Thus

$$\begin{pmatrix} \tau_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \tau_{33} \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{pmatrix} \begin{pmatrix} r_{11} & & \\ r_{12} & r_{22} & \\ r_{13} & r_{23} & r_{33} \end{pmatrix} \tag{B.19}$$

$$= \begin{pmatrix} P_{11}r_{11} + P_{12}r_{12} + P_{13}r_{13} & P_{12}r_{22} + P_{13}r_{32} & P_{13}r_{33} \\ P_{21}r_{11} + P_{22}r_{12} + P_{23}r_{13} & P_{22}r_{22} + P_{23}r_{32} & P_{23}r_{33} \\ P_{31}r_{11} + P_{32}r_{12} + P_{33}r_{13} & P_{32}r_{22} + P_{33}r_{32} & P_{33}r_{33} \end{pmatrix} , \tag{B.20}$$

and since $\boldsymbol{\tau} = \boldsymbol{\tau}^T$ and $P_{ij} = \frac{\partial \hat{\psi}}{\partial r_{ij}}$ for $i \leq j$,

$$\begin{pmatrix} \tau_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \tau_{33} \end{pmatrix} = \begin{pmatrix} \frac{\partial \hat{\psi}}{\partial r_{11}} r_{11} + \frac{\partial \hat{\psi}}{\partial r_{12}} r_{12} + \frac{\partial \hat{\psi}}{\partial r_{13}} r_{13} & \frac{\partial \hat{\psi}}{\partial r_{12}} r_{22} + \frac{\partial \hat{\psi}}{\partial r_{13}} r_{32} & \frac{\partial \hat{\psi}}{\partial r_{13}} r_{33} \\ \frac{\partial \hat{\psi}}{\partial r_{12}} r_{22} + \frac{\partial \hat{\psi}}{\partial r_{13}} r_{32} & \frac{\partial \hat{\psi}}{\partial r_{22}} r_{22} + \frac{\partial \hat{\psi}}{\partial r_{23}} r_{32} & \frac{\partial \hat{\psi}}{\partial r_{23}} r_{33} \\ \frac{\partial \hat{\psi}}{\partial r_{13}} r_{33} & \frac{\partial \hat{\psi}}{\partial r_{23}} r_{33} & \frac{\partial \hat{\psi}}{\partial r_{33}} r_{33} \end{pmatrix} \tag{B.21}$$

In particular, the matrix representation of $\boldsymbol{\tau}^S$ reads

$$\begin{pmatrix} \tau_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \tau_{33} \end{pmatrix} = \begin{pmatrix} 0 & 0 & \gamma s_1 \\ 0 & 0 & \gamma s_2 \\ 0 & 0 & f'(s_3) \end{pmatrix} \begin{pmatrix} 0 & & \\ 0 & 0 & \\ s_1 & s_2 & s_3 \end{pmatrix} \tag{B.22}$$

$$= \begin{pmatrix} \gamma s_1^2 & \gamma s_1 s_2 & \gamma s_1 s_3 \\ \gamma s_1 s_2 & \gamma s_2^2 & \gamma s_2 s_3 \\ \gamma s_1 s_3 & \gamma s_2 s_3 & f'(s_3) \end{pmatrix} \tag{B.23}$$

B.6 Frictional Contact Yield Condition

Coulomb friction places a constraint on the stress as

$$|\mathbf{t}_S| \leq -c_F \sigma_n \tag{B.24}$$

where $\sigma_n = \mathbf{a}_3^{KL} \cdot \boldsymbol{\sigma}\mathbf{a}_3^{KL}$. Recall that $\mathbf{a}_3^{KL} = \mathbf{q}_3$ and thus $\sigma_n = \mathbf{q}_3 \cdot \boldsymbol{\sigma}\mathbf{q}_3$. On the other hand, $\mathbf{t}_S$ is the tangential component of the force density and has the form $\mathbf{t}_S = (c\mathbf{q}_1 + s\mathbf{q}_2) \cdot \boldsymbol{\sigma}\mathbf{q}_3$ for some c and s such that $c^2 + s^2 = 1$. Hence, we may rewrite the constraint on stress as

$$(c\mathbf{q}_1 + s\mathbf{q}_2) \cdot \boldsymbol{\sigma}\mathbf{q}_3 + c_F\mathbf{q}_3 \cdot \boldsymbol{\sigma}\mathbf{q}_3 \leq 0. \tag{B.25}$$

Using the fact that $\boldsymbol{\sigma} = \det(\mathbf{F})\boldsymbol{\tau}$, we rewrite the constraint as

$$(c\mathbf{q}_1 + s\mathbf{q}_2) \cdot \boldsymbol{\tau}\mathbf{q}_3 + c_F\mathbf{q}_3 \cdot \boldsymbol{\tau}\mathbf{q}_3 \leq 0. \tag{B.26}$$

Substituting in the expression for $\boldsymbol{\tau}$ from equation (B.23), we find that the maximum on the left-hand-side is

$$\pm\gamma s_3\sqrt{s_1^2 + s_2^2} + c_F f' s_3$$

We apply the particular form of f in the paper where $f(x) = \frac{1}{3}k^c(1-x)^3$ for $x \leq 1$ and 0 otherwise. When $s_3 > 1$, the maximum is $\gamma s_3\sqrt{s_1^2 + s_2^2}$. In this case the return mapping set s_1 and s_2 to 0. If $0 < s_3 \leq 1$, the maximum is

$$\gamma s_3\sqrt{s_1^2 + s_2^2} - c_F k^c(s_3 - 1)^2 s_3,$$

and thus we need

$$\sqrt{s_1^2 + s_2^2} \leq \frac{c_F k^c}{\gamma}(1 - s_3)^2.$$

In this case we uniformly scale back s_1 and s_2 to satisfy the constraint.

B.7 Denting Yield Condition and Return Mapping

We apply the von Mises yield condition to the Kirchhoff-Stress in Equation (B.2)

This condition states that the deviatoric component of the stress is less than a threshold value c_{vM}

$$f_{vM}(\boldsymbol{\tau}) = |\boldsymbol{\tau} - \frac{\mathrm{tr}(\boldsymbol{\tau})}{3}\mathbf{I}|_F \leq c_{vM}. \tag{B.27}$$

This condition defines a cylindrical region of feasible states in the principal stress space since

$$f_{vM}(\boldsymbol{\tau}) = \sqrt{\frac{2}{3}\left({\tau_1}^2 + {\tau_2}^2 + {\tau_3}^2 - (\tau_1\tau_2 + \tau_2\tau_3 + \tau_1\tau_3)\right)} \tag{B.28}$$

where $\boldsymbol{\tau} = \sum_i \tau_i \mathbf{u}_i \otimes \mathbf{u}_i$ with principal stresses τ_i. The plane stress nature of $\boldsymbol{\tau}^{KL} = \sum_\alpha \tau_\alpha^{KL} \mathbf{u}_\alpha \otimes \mathbf{u}_\alpha$ means that feasible stresses are those where the principal stresses are in the ellipsoidal intersection of the cylinder and the τ_α^{KL} plane.

The yield condition is satisfied via associative projection (or return mapping) of the stress to the feasible region. The elastic and plastic strains are then computed to be consistent with the projected stress. We use $\mathbf{F}^{KL,E^{\mathrm{tr}}}, \mathbf{F}^{KL,P^{\mathrm{tr}}}$ to denote the trial state of elastoplastic strains with associated trial stress $\boldsymbol{\tau}^{KL^{\mathrm{tr}}}$. We use $\mathbf{F}^{KL,E}, \mathbf{F}^{KL,P}, \boldsymbol{\tau}^{KL}$ to denote their projected counterparts.

$$\mathbf{F}^{KL,E^{\mathrm{tr}}}, \mathbf{F}^{KL,P^{\mathrm{tr}}}, \boldsymbol{\tau}^{KL^{\mathrm{tr}}} \rightarrow \mathbf{F}^{KL,E}, \mathbf{F}^{KL,P}, \boldsymbol{\tau}^{KL}. \tag{B.29}$$

The deformation gradient constraint must be equal to the product of trial and projected elastic and plastic deformation gradients, creating the constraint on the projection

$$\mathbf{F}^{KL} = \mathbf{F}^{KL,E^{\mathrm{tr}}}\mathbf{F}^{KL,P^{\mathrm{tr}}} = \mathbf{F}^{KL,E}\mathbf{F}^{KL,P}. \tag{B.30}$$

The projection is completed by first computing the trial state of stress $\boldsymbol{\tau}^{KL^{\mathrm{tr}}}$ from $\mathbf{F}^{KL,E^{\mathrm{tr}}}$ using Equation (B.2). This is done by computing the QR decomposition of the trial elastic deformation gradient $\mathbf{F}^{KL,E^{\mathrm{tr}}} = r_{\alpha\beta}^{KL,E^{\mathrm{tr}}} \mathbf{q}_\alpha^{KL,E} \otimes \bar{\mathbf{a}}_\beta + \mathbf{q}_3^{KL,E} \otimes \bar{\mathbf{a}}_3$. Then we compute the SVD of matrix $[\mathbf{r}^{KL,E^{\mathrm{tr}}}] \in \mathbb{R}^{2\times 2}$ and the trial strain $[\boldsymbol{\epsilon}^{L^{\mathrm{tr}}}]$

$$[\mathbf{r}^{KL,E^{\mathrm{tr}}}] = [\mathbf{U}^E] \begin{pmatrix} \sigma_1^{E^{\mathrm{tr}}} & \\ & \sigma_2^{E^{\mathrm{tr}}} \end{pmatrix} [\mathbf{V}^E]^T \tag{B.31}$$

$$[\boldsymbol{\epsilon}^{L^{\mathrm{tr}}}] = [\mathbf{U}^E] \begin{pmatrix} \log(\sigma_1^{E^{\mathrm{tr}}}) & \\ & \log(\sigma_2^{E^{\mathrm{tr}}}) \end{pmatrix} [\mathbf{U}^E]^T \tag{B.32}$$

From Equation (B.2) we see that the two non-zero principal stresses ${\tau^{KL}}^{\mathrm{tr}}_{\alpha}$ of $\boldsymbol{\tau}^{KL^{\mathrm{tr}}}$ are equal to the eigenvalues of the matrix $[\boldsymbol{\tau}^{KL^{\mathrm{tr}}}]$

$$[\boldsymbol{\tau}^{KL^{\mathrm{tr}}}] = 2\mu[\boldsymbol{\epsilon}^{L^{\mathrm{tr}}}] + \lambda \mathrm{tr}([\boldsymbol{\epsilon}^{L^{\mathrm{tr}}}])\mathbf{I} = [\mathbf{U}^E] \begin{pmatrix} \tau_1^{KL^{\mathrm{tr}}} & \\ & \tau_2^{KL^{\mathrm{tr}}} \end{pmatrix} [\mathbf{U}^E]^T. \tag{B.33}$$

We therefore project the eigenvalues $({\tau^{KL}}^{\mathrm{tr}}_{\alpha} \to {\tau^{KL}}_{\alpha})$ into the ellipsoidal intersection the von Mises yield surface and the (τ_1, τ_2) plane in the direction that maximizes energy dissipation. We approximate this region by the diamond shaped region whose boundaries have slopes of ±1 to simplify the return mapping. Note that the direction of the return that maximizes energy dissipation is a function of the Cauchy-Green strain derivative of the Kirchhoff stress and thus is non-trivial to find in general. Fortunately, the quadratic Hencky strain model has the favorable property that the return direction is perpendicular to the yield surface [Mas13] which greatly simplifies the return mapping. We illustrate this property in Figure B.1. After projection, we rebuild the matrix without changing the eigenvectors and rebuild $\boldsymbol{\tau}^{KL}$ from the matrix

$$[\boldsymbol{\tau}^{KL}] = [\mathbf{U}^E] \begin{pmatrix} {\tau^{KL}}_1 & \\ & {\tau^{KL}}_2 \end{pmatrix} [\mathbf{U}^E]^T, \ \boldsymbol{\tau}^{KL} = \tau^{KL}_{\alpha\beta} \mathbf{q}_{\alpha}^{KL,E} \otimes \mathbf{q}_{\beta}^{KL,E} \tag{B.34}$$

where $\tau^{KL}_{\alpha\beta}$ are the entries in the projected matrix $[\boldsymbol{\tau}^{KL}] \in \mathbb{R}^{2\times 2}$. The projected strain $[\boldsymbol{\epsilon}^L]$

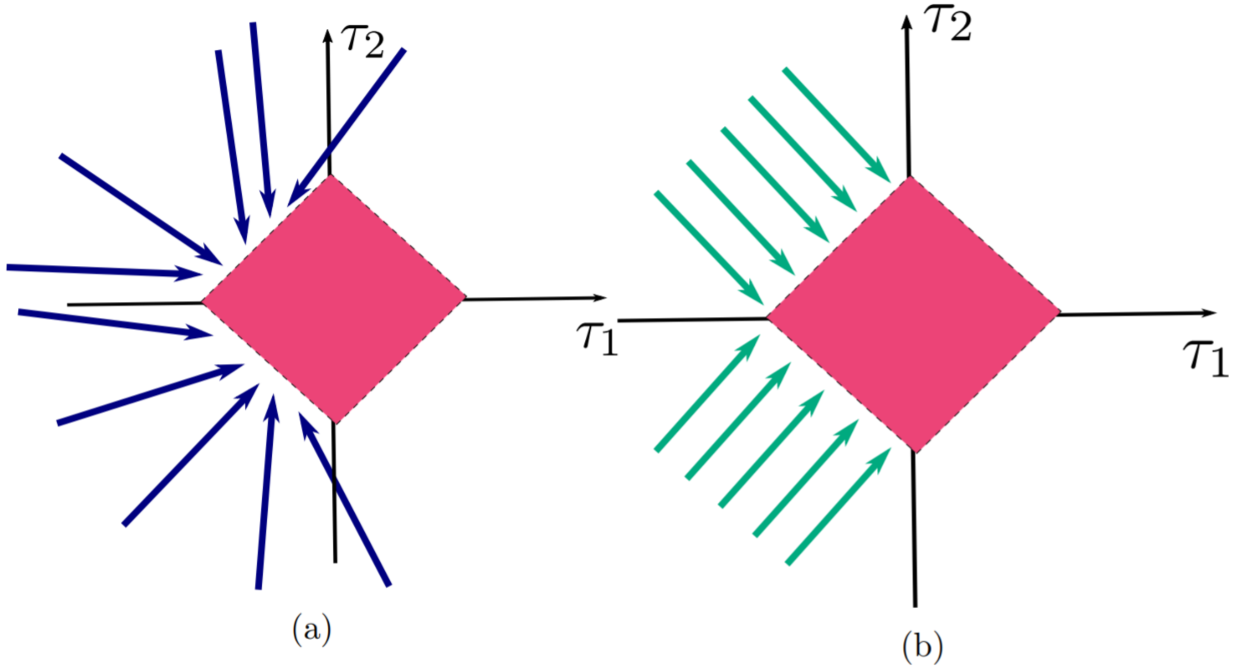

Figure B.1: **Return Mapping**. In general in the return mapping direction is non trivial (left). Quadratic Hencky strain energy density simplifies the return mapping (right).

is computed from the projected principal stresses from

$$[\boldsymbol{\epsilon}^L] = [\mathbf{U}^E] \begin{pmatrix} \log(\sigma_1^E) & \\ & \log(\sigma_2^E) \end{pmatrix} [\mathbf{U}^E]^T \tag{B.35}$$

$$\begin{pmatrix} \log(\sigma_1^E) \\ \log(\sigma_2^E) \end{pmatrix} = \begin{pmatrix} 2\mu + \lambda & \lambda \\ \lambda & 2\mu + \lambda \end{pmatrix}^{-1} \begin{pmatrix} \tau^{KL}{}_1 \\ \tau^{KL}{}_2 \end{pmatrix} \tag{B.36}$$

and the projected elastic deformation gradient is $\mathbf{F}^{KL,E} = F_{\alpha\beta}^{KL,E} \mathbf{q}_\alpha^{KL,E} \otimes \bar{\mathbf{a}}_\beta + \mathbf{q}_3^{KL,E} \otimes \bar{\mathbf{a}}_3$ where

$$[\hat{\mathbf{F}}^{KL,E}] = [\mathbf{U}^E] \begin{pmatrix} \sigma_1^E & \\ & \sigma_2^E \end{pmatrix} [\mathbf{V}^E]^T. \tag{B.37}$$

The projected plastic deformation gradient is computed from $\mathbf{F}^{KL,P} = \mathbf{F}^{KL,E^{-1}} \mathbf{F}^{KL}$ in order to maintain the constraint in Equation (B.30).

APPENDIX C

Simulation for Volumetric Objects with Frictional Contact

C.1 Cohesion-free Coupling with Traditional MPM

To prevent numerical cohesion between phases common to MPM, we adopt three separate background MPM grids, one for volumetric elastic and rigid objects, one for general MPM materials, and one for them combined. We denote quantities associated with traditional MPM particles with subscript MPM, p, quantities associated with collision particles with subscript q, quantities associated with combined grid with subscript $\mathbf{i}$, quantities associated with volumetric grid $vol, \mathbf{i}$, and quantities associated with MPM grid $MPM, \mathbf{i}$. So we have

$$m^n_{vol,\mathbf{i}} = \sum_q w^n_{\mathbf{i}q} m_q \tag{C.1}$$

$$m^n_{MPM,\mathbf{i}} = \sum_{MPM,p} w^n_{\mathbf{i}p} m_{MPM,p} \tag{C.2}$$

$$m^n_{\mathbf{i}} = m^n_{vol,\mathbf{i}} + m^n_{MPM,\mathbf{i}} \tag{C.3}$$

$$\mathbf{v}^*_{vol,\mathbf{i}} = \frac{1}{m^n_{vol,\mathbf{i}}} \sum_q w^n_{\mathbf{i}q} m_q \mathbf{v}^*_q \tag{C.4}$$

$$\mathbf{v}^n_{MPM,\mathbf{i}} = \frac{\sum_{MPM,p} w^n_{\mathbf{i}p} m_{MPM,p} \left(\mathbf{v}_{MPM,p} + \mathbf{C}_{MPM,p}(\mathbf{x}_\mathbf{i} - \mathbf{x}_{MPM,p})\right)}{m^n_{MPM,\mathbf{i}}} \tag{C.5}$$

$$\mathbf{v}^n_\mathbf{i} = \frac{m^n_{vol,\mathbf{i}} \mathbf{v}^*_{vol,\mathbf{i}} + m^n_{MPM,\mathbf{i}} \mathbf{v}^n_{MPM,\mathbf{i}}}{m^n_\mathbf{i}} \tag{C.6}$$

$$\mathbf{n}^n_\mathbf{i} = \frac{\sum_q w_{\mathbf{i}q} \mathbf{n}_q}{\|\sum_q w_{\mathbf{i}q} \mathbf{n}_q\|} \tag{C.7}$$

Grid velocity $\mathbf{v}_{MPM,\mathbf{i}}$ is updated as in [JSS15, JST16] to get $\mathbf{v}^*_{MPM,\mathbf{i}}$. Then the collision between phases is handled through an inelastic collision on collocated grid nodes. Let $\mathbf{v}_r = \mathbf{v}^*_{vol,\mathbf{i}} - \mathbf{v}^n_{MPM,\mathbf{i}}$. If $\mathbf{v}_r \cdot \mathbf{n}_\mathbf{i} < 0$, we apply impulse in the following way:

$$\mathbf{v}_t = \mathbf{v}_r - \mathbf{v}_r \cdot \mathbf{n}^n_\mathbf{i} \mathbf{n}^n_\mathbf{i} \tag{C.8}$$

$$I_\mathbf{i} = \frac{m^n_{MPM,\mathbf{i}} m^n_{vol,\mathbf{i}}}{m^n_{MPM,\mathbf{i}} + m^n_{vol,\mathbf{i}}} \mathbf{v}_r \cdot \mathbf{n}^n_q \tag{C.9}$$

$$\mathbf{v}^{n+1}_{MPM,\mathbf{i}} = \mathbf{v}^*_{MPM,\mathbf{i}} + \frac{I_\mathbf{i}\mathbf{n}_\mathbf{i}}{m^n_{MPM,\mathbf{i}}} + \min\left(-\frac{\mu I_\mathbf{i}}{m^n_{MPM,\mathbf{i}}}, \|\mathbf{v}_t\|\right) \frac{\mathbf{v}_t}{\|\mathbf{v}_t\|} \tag{C.10}$$

Finally, we interpolate the the grid velocity $\mathbf{v}^{n+1}_{MPM,\mathbf{i}}$ to MPM particles with APIC as in [JSS15, JST16], and the updated velocity of collision particle q is then

$$\mathbf{v}^\star_q = \sum_\mathbf{i} w^n_{\mathbf{i}q} \mathbf{v}^n_\mathbf{i}. \tag{C.11}$$

REFERENCES

[BFA02] R. Bridson, R. Fedkiw, and J. Anderson. “Robust Treatment of Collisions, Contact and Friction for Cloth Animation.” *ACM Trans Graph*, **21**(3):594–603, 2002.

[BLM13] T. Belytschko, W. Liu, B. Moran, and K. Elkhodary. *Nonlinear finite elements for continua and structures*. John Wiley and sons, 2013.

[BMF03] R. Bridson, S. Marino, and R. Fedkiw. “Simulation of Clothing with Folds and Wrinkles.” In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, SCA ’03, pp. 28–36, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[BR86] J. Brackbill and H. Ruppel. “FLIP: A method for adaptively zoned, Particle-In-Cell calculations of fluid flows in two dimensions.” *J Comp Phys*, **65**:314–343, 1986.

[BW08] J. Bonet and R. Wood. *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, 2008.

[CC78] Edwin Catmull and James Clark. “Recursively generated B-spline surfaces on arbitrary topological meshes.” *Computer-aided design*, **10**(6):350–355, 1978.

[CCO91] J. Collier, B. Collier, G. O’Toole, and S. Sargand. “Drape prediction by means of finite-element analysis.” *J Textile Inst*, **82**(1):96–107, 1991.

[CG95] B. Chen and M. Govindaraj. “A Physically Based Model of Fabric Drape Using Flexible Shell Theory.” *Text Res J*, **65**(6):324–330, 1995.

[Cly17] David Clyde. *Numerical Subdivision Surfaces for Simulation and Data Driven Modeling of Woven Cloth*. PhD thesis, University of California, Los Angeles, 2017.

[CO01] F. Cirak and M. Ortiz. “Fully C1-conforming subdivision elements for finite deformation thin-shell analysis.” *Int J Num Meth Eng*, **51**(7):813–833, 2001.

[COS00] F. Cirak, M. Ortiz, and P. Schröder. “Subdivision surfaces: a new paradigm for thin-shell finite-element analysis.” *Int J Num Meth Eng*, **47**(12):2039–2072, 2000.

[CTT17] D. Clyde, J. Teran, and R. Tamstorf. “Modeling and data-driven parameter estimation for woven fabrics.” In *Proc ACM SIGGRAPH / Eurograp Symp Comp Anim*, SCA ’17, pp. 17:1–17:11, New York, NY, USA, 2017. ACM.

[DB16] G. Daviet and F. Bertails-Descoubes. “A Semi-implicit Material Point Method for the Continuum Simulation of Granular Materials.” *ACM Trans Graph*, **35**(4):102:1–102:13, 2016.

[DHW19] M. Ding, X. Han, S. Wang, T. Gast, and J. Teran. "A Thermomechanical Material Point Method for Baking and Cooking." *ACM Trans Graph*, **38**(6):192:1–192:14, 2019.

[EB12] E. Edwards and R. Bridson. "A high-order accurate Particle-In-Cell method." *Int J Numer Meth Eng*, **90**:1073–1088, 2012.

[EDC96] J. Eischen, S. Deng, and T. Clapp. "Finite-element modeling and control of flexible fabric parts." *IEEE Comp Graph App*, **16**(5):71–80, 1996.

[EGS03] O. Etzmuss, J. Gross, and W. Strasser. "Deriving a particle system from continuum mechanics for the animation of deformable objects." *IEEE Trans Vis Comp Graph*, **9**(4):538–550, October 2003.

[EKS03] O. Etzmuss, M. Keckeisen, and W. Strasser. "A fast finite element solution for cloth modeling." In *Proc 11th Pac Conf Comp Graph App*, PG '03, pp. 244–254, Washington, DC, USA, 2003. IEEE Computer Society.

[EOB13] R. Echter, B. Oesterle, and M. Bischoff. "A hierarchic family of isogeometric shell finite elements." *Comp Meth App Mech Eng*, **254**:170 – 180, 2013.

[FBG18] Y. Fei, C. Batty, E. Grinspun, and C. Zheng. "A multi-scale model for simulating liquid-fabric interactions." *ACM Trans Graph*, **37**(4):51:1–51:16, 2018.

[FGG17a] C. Fu, Q. Guo, T. Gast, C. Jiang, and J. Teran. *Supplementary Technical Document*, 2017.

[FGG17b] C. Fu, Q. Guo, T. Gast, C. Jiang, and J. Teran. "A Polynomial Particle-in-cell Method." *ACM Trans Graph*, **36**(6):222:1–222:12, November 2017.

[FLL13] Y. Fan, J. Litven, D. Levin, and D. Pai. "Eulerian-on-Lagrangian Simulation." *ACM Trans Graph*, **32**(3):22:1–22:9, 2013.

[FLP14] Y. Fan, J. Litven, and D. Pai. "Active Volumetric Musculoskeletal Systems." *ACM Trans Graph*, **33**(4):152:1–152:9, 2014.

[GB17] C. Gritton and M. Berzins. "Improving accuracy in the MPM method using a null space filter." *Comp Part Mech*, **4**(1):131–142, 2017.

[GCS99] E. Grinspun, F. Cirak, P. Schröder, and M. Ortiz. "Non-linear mechanics and collisions for subdivision surfaces." *Technical report, Caltech Multi-Res Modeling Group*, 1999.

[GHD03] E. Grinspun, A. Hirani, M. Desbrun, and P. Schröder. "Discrete Shells." In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, SCA '03, pp. 62–67, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[GKS02] E. Grinspun, P. Krysl, and P. Schröder. "CHARMS: A simple framework for adaptive simulation." *ACM Trans Graph*, **21**(3):281–290, 2002.

[GLS95] L. Gan, N. Ly, and G. Steven. "A study of fabric deformation using nonlinear finite elements." *Text Res J*, **65**(11):660–668, 1995.

[GNL14] A. Golas, R. Narain, and M. Lin. "Continuum modeling of crowd turbulence." *Phys Rev E*, **90**:042816, 2014.

[GPH18] M. Gao, A. Pradhana, X. Han, Q. Guo, G. Kot, E. Sifakis, and C. Jiang. "Animating fluid sediment mixture in particle-laden flows." *ACM Trans Graph*, **37**(4):149:1–149:11, 2018.

[GS01] E. Grinspun and P. Schröder. "Normal bounds for subdivision-surface interference detection." In *IEEE Viz*, pp. 333–340, 2001.

[GS08] O. Gonzalez and A. Stuart. *A first course in continuum mechanics*. Cambridge University Press, 2008.

[GSH04] Y. Gingold, A. Secord, J. Han, E. Grinspun, and D. Zorin. "A discrete model for inelastic deformation of thin shells." In *Tech Report: Courant Institute of Mathematical Sciences, New York University*, 2004.

[GTJ17] M. Gao, A. Tampubolon, C. Jiang, and E. Sifakis. "An adaptive generalized interpolation material point method for simulating elastoplastic materials." *ACM Trans Graph*, **36**(6):223:1–223:12, 2017.

[Har64] F. Harlow. "The particle-in-cell method for numerical solution of problems in fluid dynamics." *Meth Comp Phys*, **3**:319–343, 1964.

[HN17] C. Hammerquist and J. Nairn. "A new method for material point method particle updates that reduces noise and enhances stability." *Comp Meth App Mech Eng*, **318**:724 – 738, 2017.

[HW65] F. Harlow and E. Welch. "Numerical Calculation of Time Dependent Viscous Flow of Fluid with a Free Surface." *Phys Fluid*, **8**(12):2182–2189, 1965.

[ITF04] G. Irving, J. Teran, and R. Fedkiw. "Invertible Finite Elements for Robust Simulation of Large Deformation." In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 131–140, 2004.

[JGT17] C. Jiang, T. Gast, and J. Teran. "Anisotropic elastoplasticity for cloth, knit and hair frictional contact." *ACM Trans Graph*, **36**(4), 2017.

[JSS15] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin. "The Affine Particle-In-Cell Method." *ACM Trans Graph*, **34**(4):51:1–51:10, 2015.

[JST16] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. "The Material Point Method for Simulating Continuum Materials." In *ACM SIGGRAPH 2016 Course*, pp. 24:1–24:52, 2016.

[JST17] C. Jiang, C. Schroeder, and J. Teran. "An angular momentum conserving affine-particle-in-cell method." *J Comp Phys*, **338**:137 – 164, 2017.

[KBL09] J. Kiendl, K. Bletzinger, J. Linhard, and R. Wuchner. "Isogeometric shell analysis with Kirchhoff-Love elements." *Comp Meth App Mech Eng*, **198**(49):3902 – 3914, 2009.

[KGP16] G. Klár, T. Gast, A. Pradhana, C. Fu, C. Schroeder, C. Jiang, and J. Teran. "Drucker-Prager Elastoplasticity for Sand Animation." *ACM Trans Graph*, **35**(4):103:1–103:12, 2016.

[KHW15] J. Kiendl, M. Hsu, M. Wu, and A. Reali. "Isogeometric Kirchhoff-Love shell formulations for general hyperelastic materials." *Comp Meth App Mech Eng*, **291**(Supplement C):280–303, 2015.

[KMB09] P. Kaufmann, S. Martin, M. Botsch, and M. Gross. "Implementation of discontinuous Galerkin Kirchhoff-Love shells." *ETH Zurich, Department of Computer Science, Technical Report No. 622*, 2009.

[KRO99] C. Kane, E. Repetto, M. Ortiz, and J. Marsden. "Finite element analysis of nonsmooth contact." *Comp Meth App Mech Eng*, **180**(1):1 – 26, 1999.

[LBC12] Q. Long, P. Bornemann, and F. Cirak. "Shear flexible subdivision shells." *Int J Num Meth Eng*, **90**(13):1549–1577, 2012.

[LLJ11] D. Levin, J. Litven, G. Jones, S. Sueda, and D. Pai. "Eulerian Solid Simulation with Contact." *ACM Trans Graph*, **30**(4):36:1–36:10, 2011.

[LSN13] D. Li, S. Sueda, D. Neog, and D. Pai. "Thin Skin Elastodynamics." *ACM Trans Graph*, **32**(4):49:1–49:10, 2013.

[Lu11] J. Lu. "Isogeometric contact analysis: Geometric basis and formulation for frictionless contact." *Comp Meth App Mech Eng*, **200**(5):726 – 741, 2011.

[LWH14] L. De Lorenzis, P. Wriggers, and T. Hughes. "Isogeometric contact: a review." *GAMM-Mitteilungen*, **37**(1):85–123, 2014.

[LZ14] J. Lu and C. Zheng. "Dynamic cloth simulation by isogeometric analysis." *Comp Meth App Mech Eng*, **268**(Supplement C):475 – 493, 2014.

[Mas13] C. Mast. *Modeling landslide-induced flow interactions with structures using the Material Point Method.* PhD thesis, 2013.

[MB16] M. Matzen and M. Bischoff. "A weighted point-based formulation for isogeometric contact." *Comp Meth App Mech Eng*, **308**(Supplement C):73–95, 2016.

[MCB13] M. Matzen, T. Cichosz, and M. Bischoff. "A point to segment contact formulation for isogeometric, NURBS based finite elements." *Comp Meth App Mech Eng*, **255**(Supplement C):27–39, 2013.

[MKB10] S. Martin, P. Kaufmann, M. Botsch, E. Grinspun, and M. Gross. "Unified simulation of elastic rods, shells, and solids." *ACM Trans Graph*, **29**(4):39:1–39:10, 2010.

[MS07] X. Man and C. Swan. “A mathematical modeling framework for analysis of functional clothing.” *J Eng Fibers Fabrics*, **2**(3):10–28, 2007.

[MSW09] A. McAdams, A. Selle, K. Ward, E. Sifakis, and J. Teran. “Detail Preserving Continuum Simulation of Straight Hair.” *ACM Trans Graph*, **28**(3):62:1–62:6, 2009.

[NGC09] R. Narain, A. Golas, S. Curtis, and M. Lin. “Aggregate Dynamics for Dense Crowd Simulation.” *ACM Trans Graph*, **28**(5):122:1–122:8, 2009.

[NGL10] R. Narain, A. Golas, and M. Lin. “Free-flowing granular materials with two-way solid coupling.” *ACM Trans Graph*, **29**(6):173:1–173:10, 2010.

[NPO13] R. Narain, T. Pfaff, and J. O’Brien. “Folding and crumpling adaptive sheets.” *ACM Trans Graph*, **32**(4):51:1–51:8, July 2013.

[NR08] L. Noels and R. Radovitzky. “A new discontinuous Galerkin method for Kirchhoff-Love shells.” *Comp Meth App Mech Eng*, **197**:2901–2929, 2008.

[Res16] Wolfram Research. “Mathematica 11.0.”, 2016.

[RGJ15] 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.

[SB12] 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, pp. 20:1–20:50, New York, NY, USA, 2012. ACM.

[SF89] J. Simo and D. Fox. “On a stress resultant geometrically exact shell model. Part I: Formulation and optimal parametrization.” *Comp Meth App Mech Eng*, **72**(3):267 – 304, 1989.

[SHS12] A. Stomakhin, R. Howes, C. Schroeder, and J. Teran. “Energetically consistent invertible elasticity.” In *Proc Symp Comp Anim*, pp. 25–32, 2012.

[SKB08] M. Steffen, R. Kirby, and M. Berzins. “Analysis and reduction of quadrature errors in the material point method (MPM).” *Int J Numer Meth Eng*, **76**(6):922–948, 2008.

[SSC13] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle. “A Material Point Method for snow simulation.” *ACM Trans Graph*, **32**(4):102:1–102:10, 2013.

[SSJ14] A. Stomakhin, C. Schroeder, C. Jiang, L. Chai, J. Teran, and A. Selle. “Augmented MPM for phase-change and varied materials.” *ACM Trans Graph*, **33**(4):138:1–138:11, 2014.

[Sta98] Jos Stam. "Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values." In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 395–404. ACM, 1998.

[SZS95] D. Sulsky, S. Zhou, and H. Schreyer. "Application of a particle-in-cell method to solid mechanics." *Comp Phys Comm*, **87**(1):236–252, 1995.

[TGK17] A. P. Tampubolon, T. Gast, G. Klár, C. Fu, J. Teran, C. Jiang, and K. Museth. "Multi-species simulation of porous sand and water mixtures." *ACM Trans Graph*, **36**(4), 2017.

[TLK16] Y. Teng, D. Levin, and T. Kim. "Eulerian Solid-fluid Coupling." *ACM Trans Graph*, **35**(6):200:1–200:8, 2016.

[TP16] N. Thuerey and T. Pfaff. "MantaFlow.", 2016. *http://mantaflow.com.*

[TPB87] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. "Elastically Deformable Models." *SIGGRAPH Comput Graph*, **21**(4):205–214, 1987.

[TWH11] A. Temizer, P. Wriggers, and T. Hughes. "Contact treatment in isogeometric analysis with NURBS." *Comp Meth App Mech Eng*, **200**(9):1100 – 1112, 2011.

[TWS06] B. Thomaszewski, M. Wacker, and W. Strasser. "A consistent bending model for cloth simulation with corotational subdivision finite elements." In *Proc ACM SIGGRAPH/Eurograph Symp Comp Anim*, pp. 107–116. Eurographics Association, 2006.

[UBH14] K. Um, S. Baek, and J. Han. "Advanced hybrid particle-grid method with subgrid particle correction." *Comp Graph Forum*, **33**:209–218, 2014.

[WG07] P. Wallstedt and J. Guilkey. "Improved velocity projection for the material point method." *Comp Mod in Eng and Sci*, **19**(3):223, 2007.

[WHP11] A. Wawrzinek, K. Hildebrandt, and K. Polthier. "Koiter's thin shells on Catmull-Clark limit surfaces." In Peter Eisert, Joachim Hornegger, and Konrad Polthier, editors, *Vision, Modeling, and Visualization (2011)*. The Eurographics Association, 2011.

[YSB15] Y. Yue, B. Smith, C. Batty, C. Zheng, and E. Grinspun. "Continuum foam: a material point method for shear-dependent flows." *ACM Trans Graph*, **34**(5):160:1–160:20, 2015.

[YSC18] Y. Yue, B. Smith, P. Chen, M. Chantharayukhonthorn, K. Kamrin, and E. Grinspun. "Hybrid grains: adaptive coupling of discrete and continuum simulations of granular media." *ACM Trans Graph*, **37**(6):283:1–283:19, 2018.

[ZB05] Y. Zhu and R. Bridson. "Animating sand as a fluid." *ACM Trans Graph*, **24**(3):965–972, 2005.

[ZZL17] F. Zhu, J. Zhao, S. Li, Y. Tang, and G. Wang. "Dynamically enriched MPM for invertible elasticity." In *Comp Graph Forum*, volume 36, pp. 381–392. Wiley Online Library, 2017.