Volume xx (200y), Number z, pp. 1–12

Dynamically Enriched MPM for Invertible Elasticity

Fei Zhu[1], Jing Zhao[2,3], Sheng Li[1], Yong Tang[2,3], and Guoping Wang[1]

[1]Department of Computer Science, Peking University, Beijing, China
{feizhu,lisheng,wgp}@pku.edu.cn
[2]School of Information Science and Engineering, Yanshan University, Hebei Province, China
[3]Key Laboratory of Computer Virtual Technology and System Integration, Hebei Province, China
{zhaojing,tangyong}@ysu.edu.cn

Abstract

We extend the material point method (MPM) for robust simulation of extremely large elastic deformation. This facilitates the application of MPM towards a unified solver since its versatility has been demonstrated lately with simulation of varied materials. Extending MPM for invertible elasticity requires accounting for several of its inherent limitations. MPM as a meshless method exhibits numerical fracture in large tensile deformations. We eliminate it by augmenting particles with connected material domains. Besides, constant redefinition of the interpolating functions between particles and grid introduces accumulated error which behaves like artificial plasticity. We address this problem by utilizing the Lagrangian particle domains as enriched degrees of freedom for simulation. The enrichment is applied dynamically during simulation via an error metric based on local deformation of particles. Lastly we novelly reformulate the computation in reference configuration and investigate inversion handling techniques to ensure the robustness of our method in regime of degenerated configurations. The power and robustness of our method are demonstrated with various simulations that involve extreme deformations.

Keywords: material point method, dynamical enrichment, invertible elasticity

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

The graphics community endeavors in creating impressive visual effects for animated movies and computer games. These effects vary from exaggerated squash and stretch to melting chocolate and flowing water. Specialized solvers are frequently used for specific phenomena, while a unified method to achieve these varied effects is strongly favored and remains an active research topic.

The material point method (MPM) has recently been embraced by the graphics community due to its versatility in simulating varied material behaviors. Its capability is well demonstrated with compelling visual simulations of snow [SSC*13] and other wide range of materials [SSJ*14, RGJ*15, YSB*15]. However, the simplest yet most ubiquitous hyperelastic deformation turns out to be difficult for MPM, which hinders its application as a unified simulation solver. The difficulty stems from the inherent limitations of MPM. Firstly, numerical fracture is almost inevitable because MPM is a meshless approach. Spurious tensile instability may arise when particles are separated beyond the influence range of grid cells. Secondly, MPM constantly redefines the interpolating functions between particles and the background grid during simulation, which introduces error. Inaccuracies of the deformation gradient computation accumulate as the so-called artificial plasticity. Moreover, simulating invertible elasticity introduces challenges that are more thorny than a general deformable solver because of potential degenerated configurations. Despite all these difficulties, MPM reveals the greatest potential of becoming a unified solver among existing approaches. Therefore it is worthwhile to explore extensions of MPM for possible unification.

In this paper we enhance MPM with the capability to simulate invertible elasticity, where deformations could be so wild that the determinant of deformation gradient may become negative. To our knowledge, it is the first time MPM has been extended for such extremely large deformations.

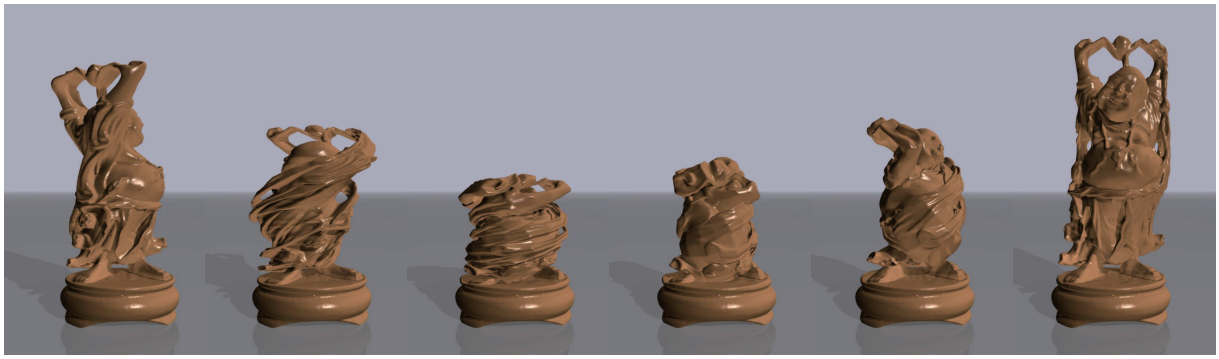

Figure 1: Buddha returns to rest state after simultaneous twisting and compression. Our enhancements extend the capability of MPM to simulate such extreme elastic deformations.

We present a number of novel contributions to achieve this goal. First, we introduce the idea of convected particle domains [BK04, SBB11, SBG13] to graphics by attaching a Lagrangian mesh with the particles. Generally there is no need for Lagrangian mesh connectivity in standard MPM, while with particle domains our method acquires several attractive features. Numerical fracture of MPM is eliminated, and we are capable of reformulating the computation in reference configuration. This reformulation combined with proper inversion handling techniques ensures robustness in degenerated deformations. The second contribution of ours is a novel enrichment strategy to reduce accumulated errors of MPM. We propose a metric to monitor the inaccuracy of particle deformation, and the simulation is enriched with the particle domains as additional degrees of freedom once the metric is satisfied. Our approach is one of the very few methods in graphical simulation [GKS02, KMB*09] that adopt the idea of enrichment to obtain additional degrees of freedom. See Figure 1 for a demonstration of our method handling extreme elastic deformations.

2. Previous Work

A complete review of the vast methods proposed thus far in computer graphics for physics-based simulation is beyond the scope of this paper. We restrict the following literature discussion to previous research on the material point method and handling of invertible elasticity.

The material point method is relatively new in computer graphics. As far as we know, Zhu and Bridson [ZB05] first mentioned it in their work without deeper investigation. Its debut as a practical technique was not made until Stomakhin et al. used it for snow simulation [SSC*13]. Subsequent works further explored its strength in simulating a broader spectrum of material behaviors [SSJ*14, RGJ*15, YSB*15]. Jiang et al. [JSS*15] recently presented a method to reduce the dissipation between particles and grid by augmenting particles with a locally affine description of velocities. In fact, graphics researchers have studied employing Eulerian grid in simulation before. McAdams et al. [MSW*09] used a cartesian grid to model hair collisions in a dynamics solver based on mass-spring systems. Levin et al. [LLJ*11] dealt with elasticity in an Eulerian framework and resolved contacts with constraints. These methods are analogous to MPM in the way of using the grid.

Although new to graphics, the analysis and applications of MPM have been extensively studied in engineering field. It was first proposed by Sulsky et al. [SCS94] as an extension of the fluid-implicit particle (FLIP) method [BR86], and substantial improvements were presented thereafter. Bardenhagen [Bar02] analyzed the energy conservation error in MPM by comparing different stress updating strategies. Steffen et al. [SKB08] studied the quadrature error and grid crossing error of MPM. They advocated the use of smoother basis functions to reduce these errors. Further investigations were carried out by Andersen and Andersen [AA10]. An alternative way to obtain a smoother field representation was the so-called generalized interpolation material point method (GIMP) [BK04], in which the notion of particle domains was first proposed. Quantities are smoothed inside the particle domains by combining the shape function of grid and the particle characteristic function. Sadeghirad and colleagues presented the convected particle domain interpolation method (CPDI) [SBB11], where particle domains are transformed as parallelograms with the local tangent affine deformation of particles. Accuracy was improved for large tensile deformations, while gaps between particles were not completely removed. They later improved accuracy to second order by tracking particle domains as quadrilaterals in 2D [SBG13]. We draw inspiration from these methods for using particle domains, but also propose novel contributions. Instead of using formulation in world space like them, we take advantage of the Lagrangian nature of particle domains and conduct reformulation in the reference configuration. Robustness in extreme deformations is ensured, and it simplifies our enrichment with inversion handling. Enrichment

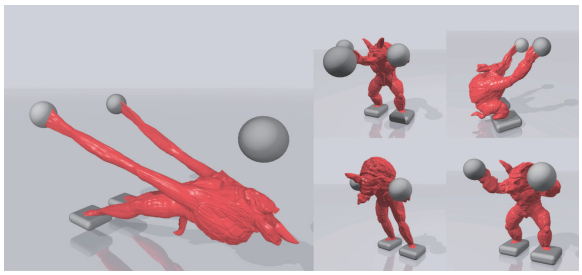

Figure 2: Armadillo is hit by a ball. MPM enhanced with particle domains prevents numerical fracture in large tensile deformations.

was also employed in [SBG13], but with a different goal of handling weak material discontinuities. Other research focuses on aspects of MPM that are less related to ours, such as contact algorithms [HZMH11, MWR14], crack growth [TN02, DLCK07], and implicit time integration [GW03, SK04].

Simulating invertible elasticity requires robust handling of physically invalid deformation mapping whose Jacobian has a negative determinant. Irving et al. [ITF04] developed the invertible finite element (IFE) framework for tetrahedral elements that extends arbitrary elastic constitutive models to inverted configurations. They later extended the method to hexahedral elements [ITF06]. Teran et al. [TSIF05] considered the difficulty in implicit time integration and presented a modified Newton-Raphson algorithm which can robustly iterate through inverted configurations. All these methods detect inversion via diagonalization of deformation gradient, and fix the invalid first Piola-Kirchhoff stresses. Stomakhin et al. [SHST12] proposed an energy-based approach that is more robust. They provide both C^1 and C^2 extensions to arbitrary isotropic energy densities. All these methods were designed for Lagrangian FEM, and we adopt the ideas in our MPM approach with Lagrangian particle domains.

3. Method Overview

The basic change we have made to standard MPM is the adoption of particle domains, which act as a middleware in the interplay of particles and background grid. The transfer of particle data to grid (equivalently referred as rasterization) can be regarded as a two-step procedure where quantities are first mapped to the corners of particle domains and then to the grid nodes, and vice versa. The key contributions of our work are built upon this seemingly simple enhancement. First, connectivity information of particle domains removes numerical fracture because initially neighbored particles interact with each other consistently through the common particle domain corners (see Figure 2). Second, data transfer between particles and domain corners is free from the accumulated error as the interpolation functions are defined in material space. This inspires us to halt the rasterization at the domain corner layer in case of severe inaccuracy and remedy the simulation by employing domain corners as additional degrees of freedom. We have designed an enrichment metric based on particle deformation and dynamically enable/disable the simulation on domain corners with this metric. Finally, the Lagrangian particle domains allow us to reformulate the computation in reference configuration, and by borrowing techniques from the invertible FEM methods we can robustly handle deformations that involve degenerated states. The full update procedure of our method is outlined in Algorithm 1, and we will describe the details in following sections.

Algorithm 1 Dynamically Enriched MPM

```
repeat
    //mark particle domain enrichment state
    for all particles do
        if enrichment metric is satisfied then
            mark its domain corners as enriched
    //rasterize
    for all particles do
        rasterize data to its domain corners
        for all domain corners of the particle do
            if domain corner is not enriched then
                rasterize data to the grid
    //solve on grid and enriched domain corners
    for all active grid nodes do
        update velocity by time integrating the dynamics
    for all enriched domain corners do
        update velocity by time integrating the dynamics,
        handle inversion during the solve
    resolve body collisions on grid nodes
    update particle deformation gradient
    //update particle velocities via two-step interpolation
    for all unenriched domain corners do
        update velocity via hybrid PIC/FLIP interpolation
        of grid velocities
    resolve body collisions on domain corners
    for all particles do
        update velocity as interpolated corner velocities
    //update domain corner positions
    for all domain corners do
        if domain corner is enriched then
            update position by time integrating velocity
        else
            update position with interpolated grid velocities
    for all particles do
        update position as interpolated corner positions
until simulation terminated
```

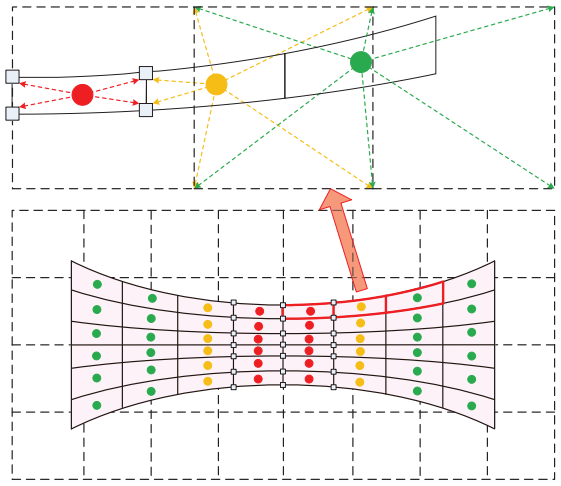

Figure 3: Illustration of MPM with particle domains in 2D (bottom). The dots in red/yellow/green stand for enriched/transitional/unenriched particles, the squares are enriched domain corners. Rasterization of quantities in the highlighted region is shown in the top of the figure.

4. MPM with Particle Domain

In standard MPM, particles interact with grid nodes as material points with concentrated mass. Better approximations can be achieved by modeling particles as material regions of a continuum. We propose to attach one particle domain with each particle, which represents the underlying portion of material. The particle domains are represented as quadrilaterals in 2D and hexahedrons in 3D. See Figure 3 for an illustration of MPM with particle domains in 2D. In this section, we defer the discussion of enrichment and describe our method without enrichment first.

Inside particle domains, alternative grid basis functions are constructed to be an interpolation of standard grid basis functions at the corners of particle domains:

$$\omega_i^*(\boldsymbol{x}) = \sum_\alpha N_\alpha^p(\boldsymbol{x})\omega_i(\boldsymbol{x}_\alpha^p). \tag{1}$$

$\omega_i(\boldsymbol{x}_\alpha^p)$ is the standard grid shape function associated with grid node i evaluated at domain corner $\boldsymbol{x}_\alpha^p$, and $N_\alpha^p(\boldsymbol{x})$ is the FEM-style shape function defined for the α^{th} corner of the particle domain evaluated at point $\boldsymbol{x}$ inside the domain.

We use the FEM shape function for linear hexahedral (quadrilateral) elements as our shape function of particle domains. We refer the readers to textbooks such as [Hug00, Bel00] for detailed explanations of such function. For the grid shape functions ω_i, we use dyadic products of one-dimensional piecewise linear functions

$$\omega_i(\boldsymbol{x}) = \omega(\frac{1}{h}(x-x_i))\omega(\frac{1}{h}(y-y_i))\omega(\frac{1}{h}(z-z_i)), \tag{2}$$

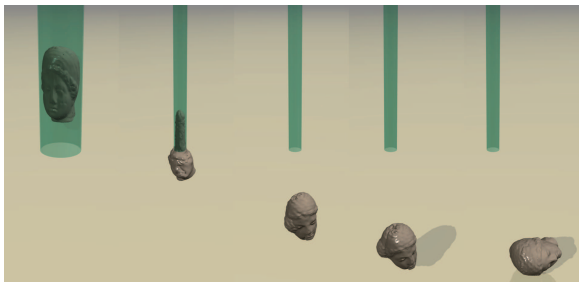

Figure 4: Egea falls through a shrinking tunnel and is compressed violently. Robustness of simulation is ensured with our reformulation in reference configuration.

where $\boldsymbol{x}_i = (x_i, y_i, z_i)$ is the position of grid node, $\boldsymbol{x} = (x, y, z)$ is the evaluation position, h is the grid spacing and

$$\omega(x) = \begin{cases} 1-|x|, & 0 \le |x| < 1 \\ 0, & \text{otherwise} \end{cases}. \tag{3}$$

Data transfer between grid and particle domain corners are performed via $\omega_i(\boldsymbol{x})$, while inside particle domains it is typical FEM approximation for linear hexahedral elements. Putting it together, particles interact with the grid using domain corners as a middleware.

As particles are in essence material regions, quantities carried by particles are computed as averages over their particle domains. Averaging over the particle domains involves spatial integration of quantities and dividing the results by volumes of the domains. This could result in numerical failure in case of degenerated configurations during simulation, e.g., flat particle domains by compression (see Figure 4). Hence we opt for the reference configuration to evaluate the result such that robustness is ensured. This differs our method from previous methods [BK04, SBB11, SBG13] that employ similar idea of particle domains. The interpolating function between particles and grid along with its gradient is evaluated as below:

$$\begin{aligned} \omega_{ip} &= \frac{1}{V_p^0}\int_{\Omega_p^0} \omega_i^*(\mathbf{X})d\Omega, \\ \nabla_{\mathbf{X}}\omega_{ip} &= \frac{1}{V_p^0}\int_{\Omega_p^0} \nabla_{\mathbf{X}}\omega_i^*(\mathbf{X})d\Omega. \end{aligned} \tag{4}$$

The gradient is computed with respect to reference coordinate $\mathbf{X}$, ω_i^* is the grid basis function inside particle domains, and Ω_p^0 is the undeformed configuration of particle domains. Replacing ω_i^* with the form in Equation 1, we get:

$$\begin{aligned} \omega_{ip} &= \frac{1}{V_p^0}\sum_\alpha \omega_i(\boldsymbol{x}_\alpha^p)\int_{\Omega_p^0} N_\alpha^p(\mathbf{X})d\Omega, \\ \nabla_{\mathbf{X}}\omega_{ip} &= \frac{1}{V_p^0}\sum_\alpha \omega_i(\boldsymbol{x}_\alpha^p)\int_{\Omega_p^0} \nabla_{\mathbf{X}}N_\alpha^p(\mathbf{X})d\Omega. \end{aligned} \tag{5}$$

The integral over undeformed particle domain averaged

by domain volume is defined as the interpolating function between particles and domain corners:

$$\omega_{\alpha p} = \frac{1}{V_p^0}\int_{\Omega_p^0} N_\alpha^p(\mathbf{X})d\Omega. \tag{6}$$

It can be precomputed, and the integral is approximated with Gauss quadrature in our implementation. We can write ω_{ip} as $\omega_{ip} = \sum_\alpha \omega_{\alpha p}\omega_i(\boldsymbol{x}_\alpha^p)$.

With interpolating functions expressed in the reference configuration, the deformation gradient for each particle is updated as:

$$\mathbf{F}_p^{n+1} = \mathbf{F}_p^n + \triangle t \nabla_{\mathbf{X}} \boldsymbol{v}_p^{n+1}, \tag{7}$$

where we have computed $\nabla_{\mathbf{X}}\boldsymbol{v}_p^{n+1} = \sum_i \boldsymbol{v}_i^{n+1}(\nabla_{\mathbf{X}}\omega_{ip}^n)^T$. Please note the difference compared with the update rule of standard MPM where formulation is constructed with respect to spatial configuration [SSC*13].

The force on grid node i resulting from elasticity is now written in terms of the first Piola-Kirchhoff stress $\mathbf{P}$ as:

$$\boldsymbol{f}_i(\boldsymbol{x}) = -\sum_p V_p^0 \mathbf{P}\nabla_{\mathbf{X}}\omega_{ip}. \tag{8}$$

Particle velocities are interpolated from grid in a hybrid PIC/FLIP manner: $\boldsymbol{v}_p^{n+1} = (1-\alpha)\boldsymbol{v}_{\mathrm{PIC}_p}^{n+1} + \alpha\boldsymbol{v}_{\mathrm{FLIP}_p}^{n+1}$, where the PIC part is $\boldsymbol{v}_{\mathrm{PIC}_p}^{n+1} = \sum_i \boldsymbol{v}_i^{n+1}\omega_{ip}$, and the FLIP part is $\boldsymbol{v}_{\mathrm{FLIP}_p}^{n+1} = \boldsymbol{v}_p^n + \sum_i(\boldsymbol{v}_i^{n+1} - \boldsymbol{v}_i^n)\omega_{ip}$. We typically used $\alpha = 0.95$. Considering the construction of ω_{ip}, this is identical to first interpolating grid velocities to particle domain corners and then to particles.

Particle domains are convected with velocities on grid if no enrichment is present. Evolving particle geometry is described by tracking locations of the particle domain corners:

$$\boldsymbol{x}_\alpha^{n+1} = \boldsymbol{x}_\alpha^n + \triangle t \sum_i \omega_i(\boldsymbol{x}_\alpha^n)\boldsymbol{v}_i^{n+1}. \tag{9}$$

$\omega_i(\boldsymbol{x}_\alpha^n)$ is the value of standard grid basis function at the α^{th} corner of particle domain. We compute new positions of particles as interpolated positions of domain corners such that particles do not drift from particle domains: $\boldsymbol{x}_p^{n+1} = \sum_\alpha \omega_{\alpha p}\boldsymbol{x}_\alpha^{n+1}$. It is worth noting that the convected domains move particles through the background grid, while the particles and corresponding domains remain fixed with each other.

4.1. Embedded Surface Meshes

Obtaining high quality rendering for particle methods is generally more challenging than mesh-based methods. Existing approaches include volume rendering [FAW10], screen space rendering [vdLGS09], *surfel* model [PKKG03, PKA*05], and meshing techniques [YT13]. Among these methods, rendering with surface meshes reconstructed from particle data is the most prevalent. While the meshing solution is widely used for liquid rendering, additional tuning such as mesh smoothing is often required. In the case of deformable objects, it is difficult to obtain a time-consistent mesh sequence with as much detail as needed.

Fortunately, particle domains equip our method with the luxury of using embedding techniques for rendering. We embed high-quality surface meshes in particle domains, and update mesh vertex positions during simulation via interpolation. The deformed surface meshes are rendered offline to generate the figures presented in this paper.

5. Dynamical Enrichment

We have not discussed enrichment thus far. In this section we will provide a detailed description of our enrichment strategy and explain the changes that have to be made with respect to the method without enrichment.

Standard MPM employs a fixed background grid as a scratch-pad of the stress-based computation. The coupling between Lagrangian particles and Eulerian grid requires re-defining interpolating functions in each time step according to their spatial positions. The data transfer between different representations leads to an inevitable loss of information due to the mismatch of resolutions. This error is reflected in the deformations and positions of the particles, and it is accumulated by constant redefinition of weight functions through the simulation. Objects could not return to rest state because of the deviations in particle positions, even if pure elastic constitutive models are employed. This behavior may be ideal for plasticity, while it must be removed for elasticity simulation. Our enrichment idea is based on the observation that shape functions of the particle domains are fixed in material space and therefore no extra error is accompanied with the interpolation inside particle domains. With this in mind, we propose to halt the interplay between particle domain corners and grid if severe inaccuracy is detected, and employ domain corners as enriched degrees of freedom. The enrichment is a dynamical process determined by a metric that we have designed to measure inaccuracy.

5.1. Enrichment Metric

We design the enrichment metric based on an intuitive assumption that more severe deformation is accompanied with more error in MPM computation. With this convention, our metric is defined as a measure of the amount of particle deformation. It is the reciprocal of the condition number of particle deformation gradient:

$$K(\mathbf{F}_p) = \frac{1}{\|\mathbf{F}_p^{-1}\|_F \cdot \|\mathbf{F}_p\|_F}, \tag{10}$$

where $\|\cdot\|_F$ is the Frobenius norm. $K(\mathbf{F}_p)$ is a scalar function of $\mathbf{F}_p$ in the range of 0 to 1, where greater values denote

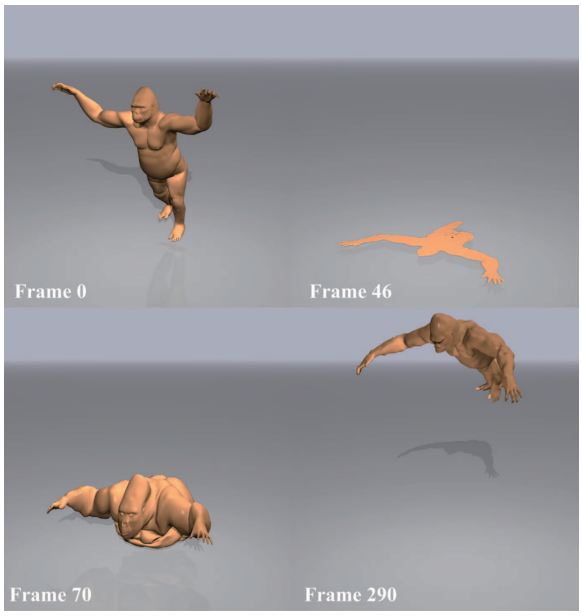

Figure 5: Gorilla collapses into flat configuration due to disabled physics (top), and quickly recovers from the degenerated state after the material strength is enabled (bottom).

better condition and vice versa. Therefore the enrichment criterion is $K(\mathbf{F}_P) \leq \varepsilon$, with ε typically set to 0.6.

It is difficult to numerically evaluate $\mathbf{F}_p^{-1}$ if $\mathbf{F}_p$ is ill-conditioned. This corresponds to extreme deformation with configurations close to degeneration. We will introduce the inversion handling procedure of our method in Section 5.3.

5.2. Simulation with Enrichment

At the beginning of each time step, we evaluate the metric function for each particle and mark all of its domain corners as enriched if the enrichment criterion is satisfied. Particles are thereby divided into three categories: enriched, transitional, and unenriched. Enriched particles are the ones with all their domain corners marked as enriched, transitional particles have both enriched and unenriched domain corners, and unenriched particles have no enriched domain corners (Figure 3). After the enrichment, the domain corners marked as enriched are also employed as computation nodes.

The dynamical enrichment alters the behavior of data transfer in our approach. During rasterization, the two-step data mapping stops at the enriched particle corners and will not continue to the grid. Enriched particles affect only the enriched particle corners, particles with no enriched corners map data to the grid in an ordinary way, and the particles in transition rasterize to both enriched domain corners and grid nodes. We illustrate the rasterization with enrichment in the top of Figure 3.

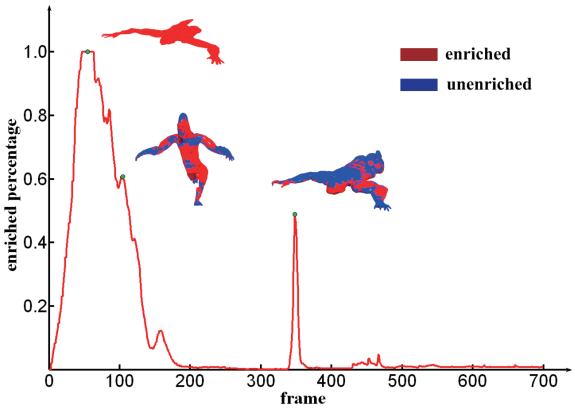

Figure 6: Percentage of enriched particles in the Gorilla example. The percentage changes dynamically during simulation. The color-mapped gorillas illustrate the distribution of enriched particles at time points denoted by green dots.

We update the velocities and positions of enriched domain corners by time integration. The mass of domain corners is precomputed by rasterizing particle masses via $\omega_{\alpha p}$ (Equation 6). Forces on the enriched domain corners are computed in the same manner with the forces on grid, replacing the corresponding weight functions. By contrast, the unenriched corners are updated through interpolation from the grid as introduced in Section 4.

In our enriched MPM, the approximation of a quantity $F(\boldsymbol{x})$ carried by the particle is described as:

$$F(\boldsymbol{x}_p) = \sum_i \omega_{ip} F_i + \sum_{\hat{\alpha}} \omega_{\hat{\alpha}p} F_{\hat{\alpha}}, \tag{11}$$

where $\hat{\alpha}$ is the enriched domain corners. As we can see, it is a generalization of our method without enrichment. The CPDI2 method [SBG13] also uses enrichment, the difference is that they use it for weak material discontinuities while we focus on reducing accumulated error due to weight function redefinition. Our dynamical enrichment metric is specially designed for this goal.

Particle velocities are still updated with the hybrid PIC/FLIP style interpolation, where both $\boldsymbol{v}_{\mathrm{PIC}_p}^{n+1}$ and $\boldsymbol{v}_{\mathrm{FLIP}_p}^{n+1}$ are computed with Equation 11. Interpreted in the other way, they are still computed via two-stage interpolations, only that velocities of enriched corners are not determined by the grid.

For particles with enriched domain corners, we compute the gradients of velocity $\nabla_{\mathbf{X}} \boldsymbol{v}_p^{n+1} = \sum_i \boldsymbol{v}_i^{n+1} (\nabla_{\mathbf{X}} \omega_{ip}^n)^T + \sum_{\hat{\alpha}} \boldsymbol{v}_{\hat{\alpha}}^{n+1} (\nabla_{\mathbf{X}} \omega_{\hat{\alpha}p}^n)^T$, and update deformation gradients with Equation 7.

Figure 5 is an example of simulation with our dynamical enrichment. Corresponding percentage of enriched particles during simulation is illustrated in Figure 6.

5.3. Inversion Handling

Since we have tackled accumulated error with enrichment, we take one step further by handling inversion in simulation. Enriched simulation on particle domains is analogous to Lagrangian FEM, and hence we could use invertible FEM techniques to handle inversion of the enriched domains. Several solutions have been presented previously for FEM [ITF04, TSIF05, ITF06, ST08, SHST12], and we choose to build on the method by Irving et al. [ITF04] for the simplicity of implementation.

In each time step we diagonalize the particle deformation gradient $\mathbf{F}_p$ via rotations $\mathbf{U}$ and $\mathbf{V}$ to obtain $\mathbf{F}_p = \mathbf{U}\hat{\mathbf{F}}_p\mathbf{V}^T$. $\hat{\mathbf{F}}_p$ is a diagonal matrix that denotes particle deformation in principal directions. An entry that is near zero corresponds to the case of flat particle domain, and negative value indicates inversion. Following Irving et al. [ITF04], we clamp the entries at some critical value if they are below that value. The first Piola-Kirchhoff stress $\mathbf{P}$ for each particle is computed as $\mathbf{P} = \mathbf{U}\hat{\mathbf{P}}\mathbf{V}^T$, where $\hat{\mathbf{P}}$ denotes the stress computed from $\hat{\mathbf{F}}_p$. $\mathbf{F}_p$ after the clamp operation is in good condition, and we use it to compute the enrichment metric $K(\mathbf{F}_p)$ without numerical failure issues.

This simple inversion handling strategy ensures that the enriched simulation on particle domains can robustly recover from inverted configurations.

6. Robust Implicit Update

We update the velocities using implicit time integration to achieve practical performance with acceptable time step. Following Stomakhin et al.'s work [SSC*13, SSJ*14] we linearize the implicit system with one step of Newton's method, which yields a (mass) symmetric system for $\boldsymbol{v}_{\tilde{i}}^{n+1}$:

$$\sum_{\tilde{j}}(\mathbf{I}\delta_{\tilde{i}\tilde{j}} + \triangle t^2 m_{\tilde{i}}^{-1}\frac{\partial^2\Phi^n}{\partial\boldsymbol{x}_{\tilde{i}}\partial\boldsymbol{x}_{\tilde{j}}})\boldsymbol{v}_{\tilde{j}}^{n+1} = \boldsymbol{v}_{\tilde{i}}^{*}, \tag{12}$$

where the right hand side is the result of explicit time integration:

$$\boldsymbol{v}_{\tilde{i}}^{*} = \boldsymbol{v}_{\tilde{i}}^{n} + \triangle t m_{\tilde{i}}^{-1}\boldsymbol{f}_{\tilde{i}}. \tag{13}$$

It is worth noting that $\tilde{i} = \{i, \hat{\alpha}\}$ here includes not only the background grid nodes i but also the enriched particle domain corners $\hat{\alpha}$. We will omit the tilde in the remainder of this section for more compact notation.

We solve the linear system in Equation 12 using the Conjugate Gradient method (CG) [She94]. Since CG is an iterative solver, we never explicitly form the coefficient matrix and instead evaluate its multiply with an arbitrary increment. The Hessian of the potential energy Φ acted on an increment $\delta\boldsymbol{u}$ is expressed as:

$$-\delta\boldsymbol{f}_i = \sum_j \frac{\partial^2\Phi}{\partial\boldsymbol{x}_i\partial\boldsymbol{x}_j}\delta\boldsymbol{u}_j = \sum_p V_p^0\mathbf{A}_p\nabla_{\mathbf{X}}\omega_{ip}, \tag{14}$$

where

$$\mathbf{A}_p = \frac{\partial^2\Phi}{\partial\mathbf{F}_p\partial\mathbf{F}_p} : (\sum_j \delta\boldsymbol{u}_j(\nabla_{\mathbf{X}}\omega_{jp})^T). \tag{15}$$

The expressions are different compared with Stomakhin et al.'s [SSC*13] due to our reformulation in material space.

The system could become indefinite in case of severe deformations and causes solver failure. It is undesirable, especially for simulations involving massive degenerated configurations. To enforce positive definiteness of the system and thereby robustness of the solving, we employ the remedy proposed by Teran et al. [TSIF05] and manipulate Equation 15 into:

$$\mathbf{A}_p = \mathbf{U}\left\{\left.\frac{\partial^2\Phi}{\partial\mathbf{F}\partial\mathbf{F}}\right|_{\hat{\mathbf{F}}_p} : \left(\mathbf{U}^T(\sum_j \delta\boldsymbol{u}_j(\nabla_{\mathbf{X}}\omega_{jp})^T)\mathbf{V}\right)\right\}\mathbf{V}^T, \tag{16}$$

where $\mathbf{U}$, $\mathbf{V}$ and $\hat{\mathbf{F}}_p$ are obtained from the diagonalization of particle deformations as introduced in Section 5.3.

We use a Jacobi preconditioner to accelerate convergence of the CG solver and thus reduce the simulation run times. The preconditioner for Equation 12 is of the form

$$\mathbf{P}_{ii} = \sum_p \mathrm{diag}(m_p\omega_{ip}\mathbf{I} + \triangle t^2 V_p^0\mathbf{H}), \tag{17}$$

where

$$\mathbf{H} = \frac{\partial^2\Phi}{\partial\mathbf{F}_p\partial\mathbf{F}_p} : (\nabla_{\mathbf{X}}\omega_{ip}(\nabla_{\mathbf{X}}\omega_{ip})^T), \tag{18}$$

and taking positive definiteness into consideration $\mathbf{H}$ now becomes

$$\mathbf{H} = \mathbf{U}\left\{\left.\frac{\partial^2\Phi}{\partial\mathbf{F}\partial\mathbf{F}}\right|_{\hat{\mathbf{F}}_p} : \left(\mathbf{U}^T(\nabla_{\mathbf{X}}\omega_{ip}(\nabla_{\mathbf{X}}\omega_{ip})^T)\mathbf{V}\right)\right\}\mathbf{V}^T. \tag{19}$$

The preconditioned solver normally converged within 20 iterations in practice. Occasionally it would take more iterations due to sudden impact of collision objects. The overall iterations reduced greatly compared with solving without a preconditioner.

7. Results

We have simulated a variety of examples that demonstrate the power of our method. All these simulations involve extreme deformations, most of which result from impact by collision objects. We process collisions with objects following Stomakhin et al. [SSC*13], except that we apply collisions to domain corners in the second phase instead of to particles. Collided domain corners are projected to the surface of collision objects in the normal direction to prevent penetration, while the impulse due to collision is applied on particles.

Figure 1 is an example of buddha model with prescribed

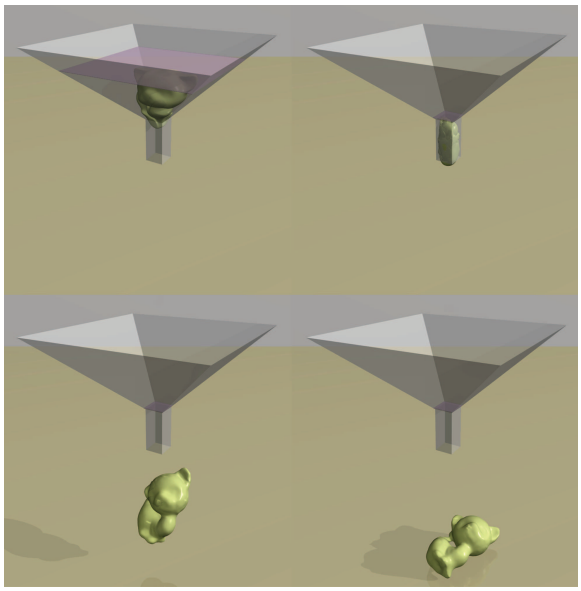

Figure 7: Kitty is forced to fall through a funnel. The hyperelastic constitutive behaviors are correctly captured by our method despite the extreme compressions.

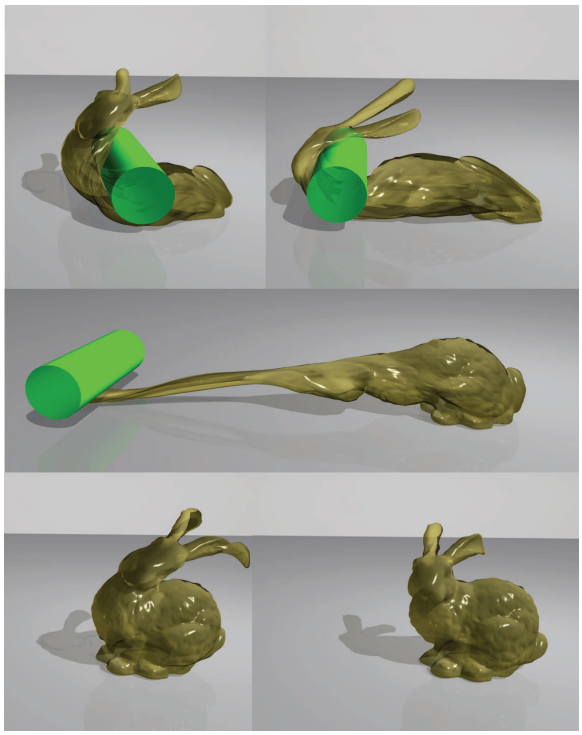

Figure 8: Bunny is rolled over by a cylinder. Our method prevents numerical fracture in the stretching and robustly handles the degenerated deformations caused by the cylinder.

twisting and compression. The buddha returns to rest state quickly after the prescribed deformation is removed. In Figure 2 we reproduce the scenario where an armadillo is hit with a ball [SHST12] and demonstrate that we can achieve similar results using MPM. Egea model in Figure 4 falls under the influence of gravity and is compressed severely in the process by a shrinking tunnel. The egea model correctly recovers after several bounces on the floor. With the example presented in Figure 5, we show that our method can handle completely flat configuration. Figure 6 explains this dynamical process by visualizing the percentage of enriched particles as a time-varying function. In Figure 7 we force a kitty to pass through a thin funnel and our method is capable of robustly handling the inversion therein. The bunny in Figure 8 is rolled over by a cylinder, and deforms with complex compression and stretching. In Figure 9 we present an example where a rabbit is sucked out of a cup through a straw. There is massive inversion while the rabbit is in the straw, and our method can handle the degeneration robustly. An illustration of the dynamical enrichment for all these examples is shown in Figure 10. We can see that the enrichment is enabled only if severe inaccuracy is detected.

Finally we demonstrate that our enriched MPM could still handle the kind of material behaviors that makes standard MPM attractive in the first place. In Figure 11 the initially elastic bunny melts into viscous phase after being heated. This is achieved by disabling the particle domains of the particles influenced by the heat source during simulation to allow topological changes between particles. We update the temperature and constitutive model parameters with a simplified implementation of Stomakhin et al.'s work [SSJ*14] without enforcing incompressibility. The surface mesh for rendering here is reconstructed from particles using the method by Bhatacharya et al. [BGB11]. In Figure 12 the armadillo is attacked from behind by a snow ball. The snow ball shatters and hits against the wall, while the armadillo deforms elastically. The different phenomena presented here are simulated with the unified MPM framework, and the coupling is handled trivially by the background grid. We use the elasto-plastic constitutive model [SSC*13] for the snow ball, and it is set artificially heavy to keep the armadillo "bowing".

Table 1 lists the simulation times and resolutions for each of the examples. Our implementation is sequential and no particular code optimization is employed. For all of the examples the grid cell size is $h = 0.5m$ and the time step size is generally $\triangle t \simeq 2.5 \times 10^{-4} s$. A less restrictive time step size $\triangle t \simeq 5.0 \times 10^{-4} s$ is used for the melting bunny example because the deformation therein is moderate. The compressible Neo-Hookean model with Young's modulus $E \simeq 5.0 \times 10^6 Pa$ and Poisson ratio $\nu = 0.3$ is employed for examples that demonstrate hyperelasticity.

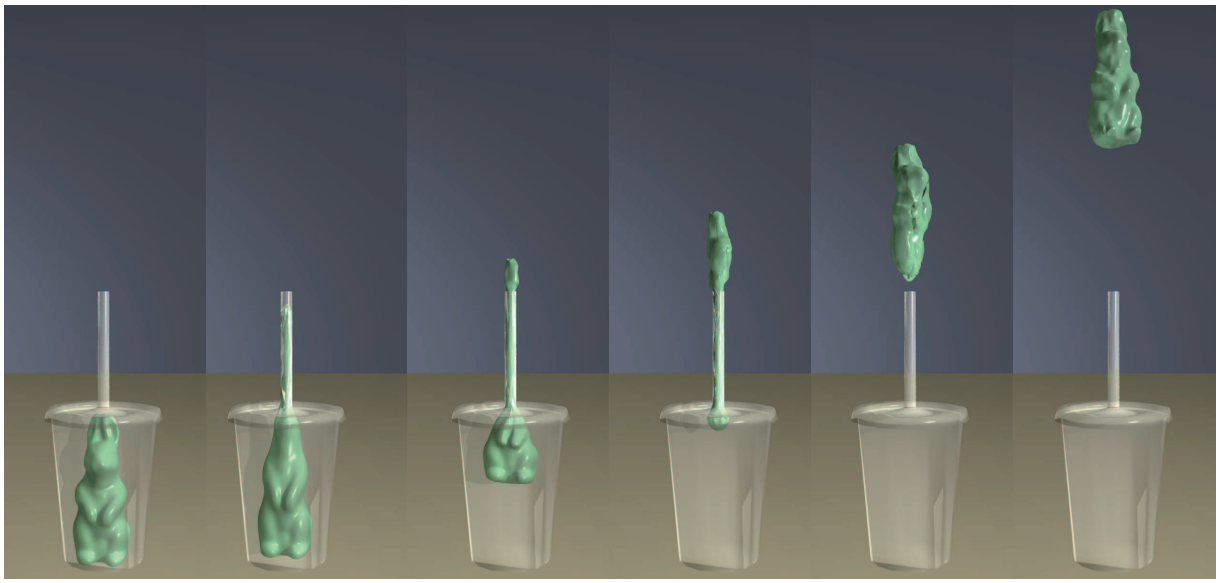

Figure 9: Rabbit is sucked out of the cup through a straw. The massive inversion and accumulated errors are well handled by our method.

Example	Particles	Grid	min/frame	Error
Buddha	2918	32×40×32	6.34	2.4e-4
Armadillo	3134	30×16×13	6.15	2.5e-3
Egea	4269	24×64×24	14.94	2.3e-5
Gorilla	4649	32×80×32	8.42	5.4e-3
Kitty	5226	96×88×96	13.85	3.7e-3
Bunny	3559	56×16×16	12.38	1.2e-3
Rabbit	2635	24×104×24	7.15	1.8e-4
Melting bunny	60000	24×22×24	15.10	N/A
Armadillo snow ball	16587	120×72×52	15.17	N/A

Table 1: Particle counts, grid resolutions, simulation times, and average particle deviations for each of our examples. The timings are measured at graphics frame rate of 30 frame/s. Simulations are performed on a single core of Intel Core i5, 2.8GHz.

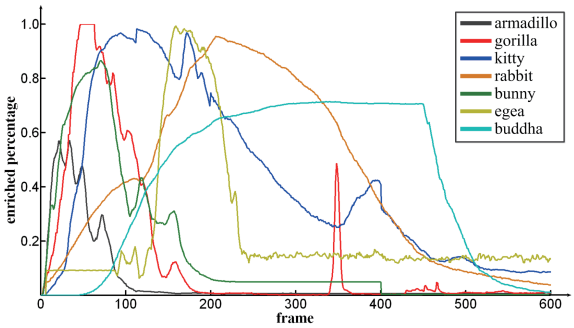

Figure 10: Illustration of dynamical enrichment with the percentage of enriched particles for the examples presented in the paper.

In order to quantitatively evaluate the efficacy of our method in reducing accumulated error, we compare the particle positions at the rest state before and after deformations. The position deviations are measured over the diagonal of the object's bounding box: $Error = \frac{|\boldsymbol{x}_p - \boldsymbol{x}_p^{ref}|}{d}$ where d is the length of the diagonal. For unconstrained motions, we first align $\boldsymbol{x}_p$ and $\boldsymbol{x}_p^{ref}$ to eliminate the rigid motions using the shape matching technique [MHTG05]. As depicted in the table, we have reduced the effect of error to a negligible magnitude.

8. Discussion and Conclusion

Comparison to FEM. MPM could be conceived as an extension of FEM, in which the computational mesh is fixed in space. The enriched particle domains are Lagrangian degrees of freedom that we use to alleviate accumulated error of standard MPM. It might seems contradictory to carry Lagrangian particle domains with MPM for hyperelasticity since FEM can easily handle such problems. We assert, however, that we switch to particle domains for additional degrees of freedom only when necessary and most of the computation is still performed on the static grid. The use of the static grid allows us to take advantage of the automatic grid-based collision handling of MPM. Besides, our modification to MPM doesn't harm the original strength of MPM because we can easily disable the particle domains during simulation if topological changes are required. It is a well-known fact that it is not trivial for FEM to

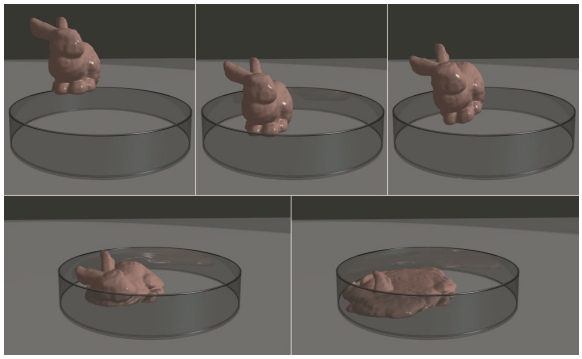

Figure 11: Bunny bounces elastically before the bottom of the container is heated (top), and melts into viscous phase after heating(bottom).

handle topological changes. Our approach combines the best aspects of both MPM and Lagrangian FEM, such that it can handle elasticity as good as FEM and at the same time retain the advantage of MPM in simulating other varied materials. In this way, multiphysics simulations can be addressed in a unified manner without the complexity of coupling between multiple methods.

Performance. Our enhancements to MPM come with some computational cost. First, the evaluation of interpolating functions is more expensive. Interpolating functions between particle domain corners and grid must be computed, as well as those between particles and grid. However, we believe this additional cost can be alleviated with parallelization as previous research has shown good scaling performance of MPM. In the future, we might consider acceleration of our method via GPGPU techniques to improve the performance. Alternatively, employing model reduction techniques [BJ05, TLP06] for MPM to reduce run times is also an interesting avenue of future work. Another source of additional cost comes from our enrichment strategy, where more computational nodes are employed. Figure 13 shows the relation between enrichment and the performance of our method with the armadillo example (Figure 2). The running time rises from 50 seconds/frame to 195 seconds/frame when the percentage of enriched particles is increased to 80% from none. Fortunately, the percentage of enrichment changes dynamically during simulation and remains at low values (below 10%) most of the time. Considering the benefits of the enrichment, we believe this cost is acceptable.

Limitations. Although we handle pure elasticity well with our dynamical enrichment, there is no warranty that the accumulated error is completely eliminated because we use a binary metric for enrichment. Increasing the enrichment threshold removes more error, but with the cost of more computation. Nevertheless, we did not encounter undesirable inaccuracy with the threshold used in our experiments (see Table 1 for the error evaluation). Another minor limitation

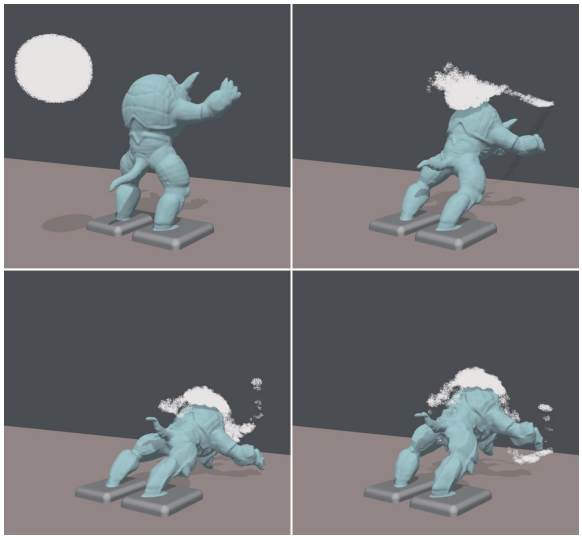

Figure 12: Armadillo hit by a snow ball. The snow ball shatters while the armadillo deforms elastically.

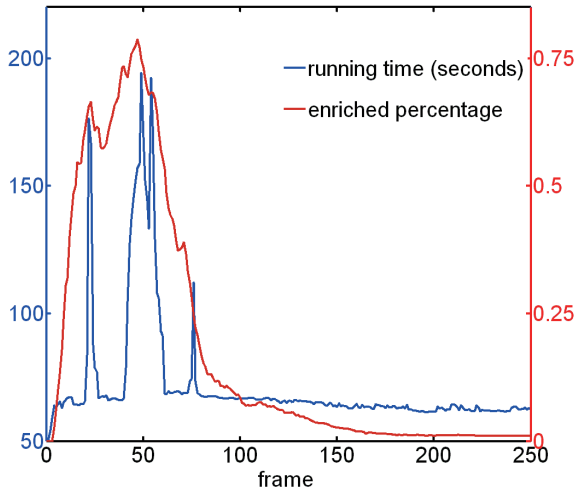

Figure 13: The running time per-frame (blue) and the percentage of enriched particles (red) for the armadillo example. The simulation time changes in accordance with the amount of enrichment and remains at acceptable low values.

of our method is that we did not handle self-collisions. The background grid handles collision well in most cases, but it can not fully resolve the penetrations due to the extremely wild deformations presented in our examples. Since it is not the focus of this paper, we leave it as future work.

Conclusion. In summary, we have introduced a novel enhancement of material point method for invertible elasticity using dynamical enrichment. The enriched

MPM is capable of robustly simulating extreme elastic deformations with degenerated configurations. Therefore, we have broadened the already wide range of materials that MPM can handle, and promoted the use of MPM as a unified solver.

Acknowledgements

We would like to thank the anonymous reviewers for the insightful suggestions. We are also grateful to Sheng Yang for sharing his mesh reconstruction code.

The authors were partially supported by Grant No. 2015BAK01B06 from The National Key Technology Research and Development Program of China, and Grant No. 61232014, 61421062, 61170205, 61472010 from National Natural Science Foundation of China.

References

[AA10] ANDERSEN S., ANDERSEN L.: Analysis of spatial interpolation in the material-point method. *Comput. Struct. 88*, 7-8 (Apr. 2010), 506–518. 2

[Bar02] BARDENHAGEN S. G.: Energy conservation error in the material point method for solid mechanics. *J. Comput. Phys. 180*, 1 (July 2002), 383–403. 2

[Bel00] BELYTSCHKO T.: *Nonlinear finite elements for continua and structures*. Wiley, Chichester New York, 2000. 4

[BGB11] BHATACHARYA H., GAO Y., BARGTEIL A.: A level-set method for skinning animated particle data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2011), SCA '11, ACM, pp. 17–24. 8

[BJ05] BARBIČ J., JAMES D. L.: Real-time subspace integration for st. venant-kirchhoff deformable models. *ACM Trans. Graph. 24*, 3 (July 2005), 982–990. 10

[BK04] BARDENHAGEN S., KOBER E.: The generalized interpolation material point method. *CMES - Computer Modeling in Engineering and Sciences 5*, 6 (2004), 477–495. 2, 4

[BR86] BRACKBILL J. U., RUPPEL H. M.: Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *J. Comput. Phys. 65*, 2 (Aug. 1986), 314–343. 2

[DLCK07] DAPHALAPURKAR N., LU H., COKER D., KOMANDURI R.: Simulation of dynamic crack growth using the generalized interpolation material point (gimp) method. *International Journal of Fracture 143*, 1 (2007), 79–102. 3

[FAW10] FRAEDRICH R., AUER S., WESTERMANN R.: Efficient high-quality volume rendering of sph data. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov. 2010), 1533–1540. 5

[GKS02] GRINSPUN E., KRYSL P., SCHRÖDER P.: Charms: A simple framework for adaptive simulation. *ACM Trans. Graph. 21*, 3 (July 2002), 281–290. 2

[GW03] GUILKEY J. E., WEISS J. A.: Implicit time integration for the material point method: Quantitative and algorithmic comparisons with the finite element method. *International Journal for Numerical Methods in Engineering 57*, 9 (2003), 1323–1338. 3

[Hug00] HUGHES T.: *The finite element method linear static and dynamic finite element analysis*. Dover Publications, Mineola, NY, 2000. 4

[HZMH11] HUANG P., ZHANG X., MA S., HUANG X.: Contact algorithms for the material point method in impact and penetration simulation. *International Journal for Numerical Methods in Engineering 85*, 4 (2011), 498–517. 3

[ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2004), SCA '04, Eurographics Association, pp. 131–140. 3, 7

[ITF06] IRVING G., TERAN J., FEDKIW R.: Tetrahedral and hexahedral invertible finite elements. *Graph. Models 68*, 2 (Mar. 2006), 66–89. 3, 7

[JSS*15] JIANG C., SCHROEDER C., SELLE A., TERAN J., STOMAKHIN A.: The affine particle-in-cell method. *ACM Trans. Graph. 34*, 4 (July 2015), 51:1–51:10. 2

[KMB*09] KAUFMANN P., MARTIN S., BOTSCH M., GRINSPUN E., GROSS M.: Enrichment textures for detailed cutting of shells. *ACM Trans. Graph. 28*, 3 (July 2009), 50:1–50:10. 2

[LLJ*11] LEVIN D. I. W., LITVEN J., JONES G. L., SUEDA S., PAI D. K.: Eulerian solid simulation with contact. *ACM Trans. Graph. 30*, 4 (July 2011), 36:1–36:10. 2

[MHTG05] MÜLLER M., HEIDELBERGER B., TESCHNER M., GROSS M.: Meshless deformations based on shape matching. *ACM Trans. Graph. 24*, 3 (July 2005), 471–478. 9

[MSW*09] MCADAMS A., SELLE A., WARD K., SIFAKIS E., TERAN J.: Detail preserving continuum simulation of straight hair. *ACM Trans. Graph. 28*, 3 (July 2009), 62:1–62:6. 2

[MWR14] MA J., WANG D., RANDOLPH M.: A new contact algorithm in the material point method for geotechnical simulations. *International Journal for Numerical and Analytical Methods in Geomechanics 38*, 11 (2014), 1197–1210. 3

[PKA*05] PAULY M., KEISER R., ADAMS B., DUTRÉ P., GROSS M., GUIBAS L. J.: Meshless animation of fracturing solids. *ACM Trans. Graph. 24*, 3 (July 2005), 957–964. 5

[PKKG03] PAULY M., KEISER R., KOBBELT L. P., GROSS M.: Shape modeling with point-sampled geometry. *ACM Trans. Graph. 22*, 3 (July 2003), 641–650. 5

[RGJ*15] RAM D., GAST T., JIANG C., SCHROEDER C., STOMAKHIN A., TERAN J., KAVEHPOUR P.: A material point method for viscoelastic fluids, foams and sponges. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (New York, NY, USA, 2015), SCA '15, ACM, pp. 157–163. 1, 2

[SBB11] SADEGHIRAD A., BRANNON R. M., BURGHARDT J.: A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations. *International Journal for Numerical Methods in Engineering 86*, 12 (2011), 1435–1456. 2, 4

[SBG13] SADEGHIRAD A., BRANNON R., GUILKEY J.: Second-order convected particle domain interpolation (cpdi2) with enrichment for weak discontinuities at material interfaces. *International Journal for Numerical Methods in Engineering 95*, 11 (2013), 928–952. 2, 3, 4, 6

[SCS94] SULSKY D., CHEN Z., SCHREYER H.: A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering 118*, 1-2 (1994), 179 – 196. 2

[She94] SHEWCHUK J. R.: *An Introduction to the Conjugate*

Gradient Method Without the Agonizing Pain. Tech. rep., Pittsburgh, PA, USA, 1994. 7

[SHST12] STOMAKHIN A., HOWES R., SCHROEDER C., TERAN J. M.: Energetically consistent invertible elasticity. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2012), SCA '12, Eurographics Association, pp. 25–32. 3, 7, 8

[SK04] SULSKY D., KAUL A.: Implicit dynamics in the material-point method. *Computer Methods in Applied Mechanics and Engineering 193*, 12-14 (2004), 1137 – 1170. Meshfree Methods: Recent Advances and New Applications. 3

[SKB08] STEFFEN M., KIRBY R. M., BERZINS M.: Analysis and reduction of quadrature errors in the material point method (mpm). *International Journal for Numerical Methods in Engineering 76*, 6 (2008), 922–948. 2

[SSC*13] STOMAKHIN A., SCHROEDER C., CHAI L., TERAN J., SELLE A.: A material point method for snow simulation. *ACM Trans. Graph. 32*, 4 (July 2013), 102:1–102:10. 1, 2, 5, 7, 8

[SSJ*14] STOMAKHIN A., SCHROEDER C., JIANG C., CHAI L., TERAN J., SELLE A.: Augmented mpm for phase-change and varied materials. *ACM Trans. Graph. 33*, 4 (July 2014), 138:1–138:11. 1, 2, 7, 8

[ST08] SCHMEDDING R., TESCHNER M.: Inversion handling for stable deformable modeling. *The Visual Computer 24*, 7-9 (2008), 625–633. 7

[TLP06] TREUILLE A., LEWIS A., POPOVIĆ Z.: Model reduction for real-time fluids. *ACM Trans. Graph. 25*, 3 (July 2006), 826–834. 10

[TN02] TAN H., NAIRN J. A.: Hierarchical, adaptive, material point method for dynamic energy release rate calculations. *Computer Methods in Applied Mechanics and Engineering 191* (2002), 2123–2137. 3

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2005), SCA '05, ACM, pp. 181–190. 3, 7

[vdLGS09] VAN DER LAAN W. J., GREEN S., SAINZ M.: Screen space fluid rendering with curvature flow. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2009), I3D '09, ACM, pp. 91–98. 5

[YSB*15] YUE Y., SMITH B., BATTY C., ZHENG C., GRINSPUN E.: Continuum foam: A material point method for shear-dependent flows. *ACM Trans. Graph. 34*, 5 (Nov. 2015), 160:1–160:20. 1, 2

[YT13] YU J., TURK G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Trans. Graph. 32*, 1 (Feb. 2013), 5:1–5:12. 5

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph. 24*, 3 (July 2005), 965–972. 2