Material point method, an almost complete walkthrough[*]

Zhuo Lu (Seth)

March 22, 2019

Contents

1 Introduction

This document springs out from my half-semester project for COMPSCI 284B, Spring 2019 at UC Berkeley to make a baseline snow simulation. It should compile existing references from published literature fairly completely. I guess this may serve as a useful resource and somewhat complete reference for those who're genuinely confused when first attempting to implement the iteration of material point method published by Stomakhin et al. [5] However, I do recommend trying to study material point method by diving into the the literature first before using this document as the only source of input for this subject.

This document is written to be rather exhaustive in stepping through the derivations that are often left out in published literature. However, it is not written with explaining the governing physics equations. An implementation following the full method in this document is available on GitHub for reference: `https://github.com/sethlu/renderbox-snow`.

[*]Please be aware that this document is not yet fully proofread.

2 Prior work

Most of the materials compiled in this document reference existing literature and open-sourced implementations.

Below are a few open-sourced implementations:

- `https://github.com/Azmisov/snow`
- `https://github.com/yuanming-hu/taichi_mpm`
- `https://github.com/WindQAQ/MPM`

3 Notation

Below are a few physical quantities are frequently used in various equations. Non-scalars should be consistently **bolded** throughout the document.

- Mass m
- Velocity $\boldsymbol{v}$
- Displacement $\boldsymbol{x} = (x, y, z)$
- Time t
- Time step Δt
- Volume V
- Density ρ
- Force $\boldsymbol{f}$
- Deformation gradient $\boldsymbol{F}$
- Grid spacing h

Some of those quantities may relate to a grid node or a particle node. If subject to a grid node, the symbol will be subscripted with $\square_{\boldsymbol{i}}$, where $\boldsymbol{i} = (i, j, k)$. Other times it may be subscripted with $\square_{\boldsymbol{j}}$ to denote a different grid node. If subject to a particle node, the symbol will be subscripted with $\square_p$.

Some quantities vary over time: A variable at time step n will be superscripted as $\square^n$. Similarly, a variable at time step $(n+1)$ will be superscripted as $\square^{n+1}$. For instance, the volume for a particle at time 0 is notated as V_p^0.

A grid node location at time step n is referred to as $\boldsymbol{x}_{\boldsymbol{i}}^n$; at time step $(n+1)$ it is often referred to as $\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x}_{\boldsymbol{i}}^{n+1}$, representing a deformed location for the grid node. Later we will mention this but the grid node locations remain unchanged even though we talk about deformations on the grid node.

For a matrix $\boldsymbol{A}$, its entries are referred to as $\boldsymbol{A}_{ij}$ in a column-major order, i as the column and j as the row (zero-indexed).

4 Building blocks

4.1 Grid basis function

The one-dimensional B-splines function is defined as follows (from Stomakhin et al. [5])

$$N(x) = \begin{cases} \frac{1}{2}|x|^3 - x^2 + \frac{2}{3} & 0 \le |x| < 1 \\ -\frac{1}{6}|x|^3 + x^2 - 2|x| + \frac{4}{3} & 1 \le |x| < 2 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The derivative of the one-dimensional B-splines function is then

$$\delta N(x) = \begin{cases} \frac{1}{2}x^2 - 2|x| + 2 & -2 \le x < -1 \\ -\frac{3}{2}x^2 + 2|x| & -1 \le x < 0 \\ \frac{3}{2}x^2 - 2|x| & 0 \le x < 1 \\ -\frac{1}{2}x^2 + 2|x| - 2 & 1 \le x < 2 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Moving to three dimensions, the B-splines function is defined for each grid-particle pair as

$$N_{\boldsymbol{i}}(\mathbf{x}_p) = N(\frac{1}{h}(x_p - ih))N(\frac{1}{h}(y_p - jh))N(\frac{1}{h}(z_p - kh)) \tag{3}$$

$$= N(\frac{1}{h}(x_p - x_{\boldsymbol{i}}))N(\frac{1}{h}(y_p - y_{\boldsymbol{i}}))N(\frac{1}{h}(z_p - z_{\boldsymbol{i}})) \tag{4}$$

The gradient $\nabla N_{\boldsymbol{i}}(\boldsymbol{x}_p)$ can be computed as (from Jiang et al. [4])

$$\nabla N_{\boldsymbol{i}}(\boldsymbol{x}_p) = \begin{bmatrix} \frac{1}{h}\delta N(\frac{1}{h}(x_p - ih))N(\frac{1}{h}(y_p - jh))N(\frac{1}{h}(z_p - kh)) \\ N(\frac{1}{h}(x_p - ih))\frac{1}{h}\delta N(\frac{1}{h}(y_p - jh))N(\frac{1}{h}(z_p - kh)) \\ N(\frac{1}{h}(x_p - ih))N(\frac{1}{h}(y_p - jh))\frac{1}{h}\delta N(\frac{1}{h}(z_p - kh)) \end{bmatrix} \tag{5}$$

We define the weight and its gradient as (from Stomakhin et al. [5])

$$w_{\boldsymbol{i}p} = N_{\boldsymbol{i}}(\boldsymbol{x}_p) \tag{6}$$

$$\nabla w_{\boldsymbol{i}p} = \nabla N_{\boldsymbol{i}}(\boldsymbol{x}_p) \tag{7}$$

4.2 Elastic-plastic energy density function

Given elastic and plastic deformations, $\boldsymbol{F}_E$ and $\boldsymbol{F}_P$, the elastic-plastic energy density function is defined as (from Stomakhin et al. [5])

$$\Psi(\boldsymbol{F}_E, \boldsymbol{F}_P) = \mu||\boldsymbol{F}_E - \boldsymbol{R}_E||_F^2 + \frac{\lambda}{2}(J_E - 1)^2 \tag{8}$$

where

$$\mu = \mu_0 e^{\xi(1-J_P)}$$
$$\lambda = \lambda_0 e^{\xi(1-J_P)}$$
$$J_E = \det \boldsymbol{F}_E$$
$$J_P = \det \boldsymbol{F}_P$$

The term $\boldsymbol{R}_E$ is obtained by polar decomposition of $\boldsymbol{F}_E = \boldsymbol{R}_E\boldsymbol{S}_E$. The μ_0 and λ_0 are the initial Lamé parameters. And ξ is the hardening parameter.

Partial derivative of energy density function

The partial derivative with respect to elastic deformation is (from Stomakhin et al. [6])

$$\frac{\partial \Psi}{\partial \boldsymbol{F}_E}(\boldsymbol{F}_E, \boldsymbol{F}_P) = 2\mu(\boldsymbol{F}_E - \boldsymbol{R}_E) + \lambda(J_E - 1)J_E\boldsymbol{F}_E^{-T} \tag{9}$$

4.3 Internal stress-based forces

In this section, $\hat{\boldsymbol{x}}$ refers to $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ of all the grid nodes.

Given an elasto-plastic energy density function $\Psi(\boldsymbol{F}_E, \boldsymbol{F}_P)$, we can evaluate it at each particle node as $\Psi_p = \Psi(\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p})$ using the elastic and plastic parts of the particle deformation gradient $\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}})$ and $\boldsymbol{F}^n_{P_p}$. The deformed locations of all grid nodes are represented as $\hat{\boldsymbol{x}}$.

The MPM approximation of the total elastic potential relates to $\hat{\boldsymbol{x}}$ as (from Stomakhin et al. [5])

$$\Phi(\hat{\boldsymbol{x}}) = \sum_p V_p^0 \Psi(\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p}) \tag{10}$$

where the elastic deformation gradient relates to $\hat{\boldsymbol{x}}$ as (from Stomakhin et al. [5])

$$\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}) = (\boldsymbol{I} + \sum_{\boldsymbol{i}} (\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x}^n_{\boldsymbol{i}})(\nabla w_{\boldsymbol{i}p})^T)\boldsymbol{F}^n_{E_p} \tag{11}$$

As an extra note, we can also talk about $\delta \boldsymbol{F}_{E_p}$ over some increment $\delta \boldsymbol{x}_{\boldsymbol{i}}$

$$\delta \boldsymbol{F}_{E_p} = \sum_{\boldsymbol{i}} \delta \boldsymbol{x}_{\boldsymbol{i}} (\nabla w_{\boldsymbol{i}p})^T \boldsymbol{F}^n_{E_p} \tag{12}$$

The stress-based force at each grid node is given as (from Stomakhin et al. [5])

$$\begin{aligned} \boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}}) &= -\frac{\partial \Phi}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}}}(\hat{\boldsymbol{x}}) \\ &= -\sum_p V_p^0 \frac{\partial \Psi}{\partial \boldsymbol{F}_E}(\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p})(\boldsymbol{F}^n_{E_p})^T \nabla w^n_{\boldsymbol{i}p} \\ &= -\sum_p V_p^n \boldsymbol{\sigma}_p \nabla w^n_{\boldsymbol{i}p} \end{aligned} \tag{13}$$

where the Cauchy stress $\boldsymbol{\sigma}_p = \frac{1}{J_P^n} \frac{\partial \Psi}{\partial \boldsymbol{F}_E}(\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p})(\boldsymbol{F}^n_{E_p})^T$.

Force derivative for semi-implicit integration

Please find the derivations in the semi-implicit integration section.

5 Full method

Unless otherwise specified, the update method is based off the paper by Stomakhin et al. [5] Some equations are expanded for ease of reference.

5.1 Rasterize particle data to grid

The particle mass and velocity are transferred to the grid.

The grid mass $m_{\boldsymbol{i}}$ is computed as

$$m_{\boldsymbol{i}}^n = \sum_p m_p w_{\boldsymbol{i}p}^n \tag{14}$$

The grid velocity $\boldsymbol{v}_{\boldsymbol{i}}$ can be computed with the following, as weighted by grid mass

$$\boldsymbol{v}_{\boldsymbol{i}} = \frac{\sum_p \boldsymbol{v}_p^n m_p w_{\boldsymbol{i}p}^n}{m_{\boldsymbol{i}}^n} \tag{15}$$

5.2 Compute particle volumes (first time only)

The initial particle volume V_p^0 is computed as

$$V_p^0 = \frac{m_p}{\rho_p^0} \tag{16}$$

where the density for particle ρ_p^0 is estimated as (assuming constant grid node volume h^3)

$$\rho_p^0 = \frac{\sum_{\boldsymbol{i}} m_{\boldsymbol{i}}^0 w_{\boldsymbol{i}p}^0}{h^3} \tag{17}$$

5.3 Compute grid forces

Recall that the internal stress-based force per grid node is defined as

$$\boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}}) = -\sum_p V_p^0 \frac{\partial \Psi}{\partial \boldsymbol{F}_E}(\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}_{P_p}^n)(\boldsymbol{F}_{E_p}^n)^T \nabla w_{\boldsymbol{i}p}^n$$

Also recall the elastic deformation gradient

$$\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}) = (\boldsymbol{I} + \sum_{\boldsymbol{i}} (\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x}_{\boldsymbol{i}}^n)(\nabla w_{\boldsymbol{i}p})^T)\boldsymbol{F}_{E_p}^n$$

Since the grid nodes does not move (even under deformation), we can set $\hat{\boldsymbol{x}} = \boldsymbol{x}^n$. Then we have

$$\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}) = \boldsymbol{F}_{E_p}^n \tag{18}$$

Then we can compute the internal force

$$\begin{aligned}
\boldsymbol{f}_{\boldsymbol{i}}^n &= \boldsymbol{f}_{\boldsymbol{i}}(\boldsymbol{x}^n) \\
&= -\sum_p V_p^0 \frac{\partial \Psi}{\partial \boldsymbol{F}_E}(\boldsymbol{F}_{E_p}^n, \boldsymbol{F}_{P_p}^n)(\boldsymbol{F}_{E_p}^n)^T \nabla w_{\boldsymbol{i}p}^n \\
&= -\sum_p V_p^0 (2\mu(\boldsymbol{F}_{E_p}^n - \boldsymbol{R}_{E_p}^n) + \lambda(J_{E_p} - 1)J_{E_p}(\boldsymbol{F}_{E_p}^n)^{-T})(\boldsymbol{F}_{E_p}^n)^T \nabla w_{\boldsymbol{i}p}^n \\
&= -\sum_p V_p^0 (2\mu(\boldsymbol{F}_{E_p}^n - \boldsymbol{R}_{E_p}^n)(\boldsymbol{F}_{E_p}^n)^T + \lambda(J_{E_p} - 1)J_{E_p}\boldsymbol{I})\nabla w_{\boldsymbol{i}p}^n
\end{aligned} \tag{19}$$

where

$$\mu = \mu_0 e^{\xi(1-J_{P_p})}$$
$$\lambda = \lambda_0 e^{\xi(1-J_{P_p})}$$
$$J_{E_p} = \det \boldsymbol{F}_{E_p}$$
$$J_{P_p} = \det \boldsymbol{F}_{P_p}$$

The term $\boldsymbol{R}_{E_p}$ is obtained by polar decomposition of $\boldsymbol{F}_{E_p} = \boldsymbol{R}_{E_p}\boldsymbol{S}_{E_p}$. The μ_0 and λ_0 are the initial Lamé parameters. And ξ is the hardening parameter.

Then we can combine the internal force $\boldsymbol{f}_{\boldsymbol{i}}^n$ with some external force $\boldsymbol{g}$ to find the net force acting on the grid node. Here we assume $\boldsymbol{g}$ is constant so it won't affect the gradient taken on the net force over the grid.

5.4 Update velocities on grid

The grid velocity is computed as

$$\boldsymbol{v}_{\boldsymbol{i}}^* = \boldsymbol{v}_{\boldsymbol{i}}^n + \Delta t m_{\boldsymbol{i}}^{-1}(\boldsymbol{f}_{\boldsymbol{i}}^n + \boldsymbol{g}) \tag{20}$$

5.5 Grid-based body collisions

Refer to the collision handling section below to update $\boldsymbol{v}_{\boldsymbol{i}}^*$.

5.6 Solve the linear system

With explicit Euler integration, the velocity is updated as

$$\boldsymbol{v}_{\boldsymbol{i}}^{n+1} = \boldsymbol{v}_{\boldsymbol{i}}^* \tag{21}$$

For semi-implicit integration, refer to the section below.

5.7 Update deformation gradient

The deformation gradient for a particle (elastic-plastic combined) is updated as

$$\boldsymbol{F}_p^{n+1} = (\boldsymbol{I} + \Delta t \nabla \boldsymbol{v}_p^{n+1})\boldsymbol{F}_{E_p}^n \boldsymbol{F}_{P_p}^n = \boldsymbol{F}_{E_p}^{n+1}\boldsymbol{F}_{P_p}^{n+1} \tag{22}$$

where the velocity gradient

$$\nabla \boldsymbol{v}_p^{n+1} = \sum_{\boldsymbol{i}} \boldsymbol{v}_{\boldsymbol{i}}^{n+1}(\nabla w_{\boldsymbol{i}p}^n)^T \tag{23}$$

Initially, all changes are attributed to the elastic part (the hat-symbol here represents a temporary value)

$$\hat{\boldsymbol{F}}_{E_p}^{n+1} = (\boldsymbol{I} + \Delta t \nabla \boldsymbol{v}_p^{n+1})\boldsymbol{F}_{E_p}^n$$

Since some parts of $\boldsymbol{F}_{E_p}^{n+1}$ may exceed the critical deformation threshold, the singular value decomposition is computed for

$$\hat{\boldsymbol{F}}_{E_p}^{n+1} = \boldsymbol{U}_p \hat{\boldsymbol{\Sigma}}_p \boldsymbol{V}_p^T$$

Then each of the entries of $\boldsymbol{\Sigma}_p$ is clamped so it's within the critical compression/stretch range $\boldsymbol{\Sigma}_p = \text{clamp}(\hat{\boldsymbol{\Sigma}}_p, [1-\theta_c, 1+\theta_s])$. The clamped $\boldsymbol{\Sigma}_p$ is then used to construct the final elastic deformation for the particle

$$\boldsymbol{F}_{E_p}^{n+1} = \boldsymbol{U}_p \boldsymbol{\Sigma}_p \boldsymbol{V}_p^T \tag{24}$$

The final plastic deformation for the particle is computed as

$$\boldsymbol{F}_{P_p}^{n+1} = (\boldsymbol{F}_{E_p}^{n+1})^{-1}\boldsymbol{F}_p^{n+1} = \boldsymbol{V}_p \boldsymbol{\Sigma}_p^{-1} \boldsymbol{U}_p^T \boldsymbol{F}_p^{n+1} \tag{25}$$

5.8 Update particle velocities

The particle velocity is updated with part FLIP and part PIC [5]

$$\boldsymbol{v}_p^{n+1} = (1-\alpha)\boldsymbol{v}_{\mathrm{PIC}_p}^{n+1} + \alpha \boldsymbol{v}_{\mathrm{FLIP}_p}^{n+1} \tag{26}$$

$$\boldsymbol{v}_{\mathrm{PIC}_p}^{n+1} = \sum_{\boldsymbol{i}} \boldsymbol{v}_{\boldsymbol{i}}^{n+1} w_{\boldsymbol{i}p}^{n} \tag{27}$$

$$\boldsymbol{v}_{\mathrm{FLIP}_p}^{n+1} = \boldsymbol{v}_p^{n} + \sum_{\boldsymbol{i}} (\boldsymbol{v}_{\boldsymbol{i}}^{n+1} - \boldsymbol{v}_{\boldsymbol{i}}^{n}) w_{\boldsymbol{i}p}^{n} \tag{28}$$

5.9 Particle-based body collisions

Refer to the collision handling section below to update $\boldsymbol{v}_p^{n+1}$.

5.10 Update particle positions

The particle positions are updated as

$$\boldsymbol{x}_p^{n+1} = \boldsymbol{x}_p^{n} + \Delta t \boldsymbol{v}_p^{n+1} \tag{29}$$

Up to this point, your simulation should look fairly decent, though without external collisions. However, with the explicit update step, the stiffness from the internal forces is very likely to blow up at larger time steps.

6 Filling in the missing pieces

6.1 Collision handling

From Stomakhin et al. [5], given node velocity $\boldsymbol{v}$, its relative velocity v_{rel} to the collider object v_{co} is computed as

$$\boldsymbol{v}_{rel} = \boldsymbol{v} - \boldsymbol{v}_{co} \tag{30}$$

Given the surface normal of the collider object $\boldsymbol{n}$ (unit length) we can compute

$$v_n = \boldsymbol{v}_{rel} \cdot \boldsymbol{n} \tag{31}$$

If $v_n \geq 0$, we could tell the node is moving away from the colliding surface as it's relatively moving in the direction of the surface normal. In that way, there is no collision.

We can compute the tangential portion of the relative velocity as

$$\boldsymbol{v}_t = \boldsymbol{v}_{rel} - v_n \boldsymbol{n} \tag{32}$$

When $||\boldsymbol{v}_t|| \leq -\mu v_n$, we apply sticking impulse by setting

$$\boldsymbol{v}'_{rel} = \boldsymbol{0} \tag{33}$$

Otherwise, we apply dynamic friction and set

$$\boldsymbol{v}'_{rel} = \boldsymbol{v}_t + \mu v_n \frac{\boldsymbol{v}_t}{||\boldsymbol{v}_t||} \tag{34}$$

We can also set $\boldsymbol{v}'_{rel} = \boldsymbol{0}$ unconditionally when wanting to have the snow stick to vertical or underhanging surfaces.

The velocity due to collision is finally updated as

$$\boldsymbol{v}' = \boldsymbol{v}'_{rel} + \boldsymbol{v}_{co} \tag{35}$$

6.2 Semi-implicit integration

Recall that $\hat{\boldsymbol{x}} = \boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \Delta t \boldsymbol{v}^{n+1}$ and $\boldsymbol{v}^*$ is the velocity from explicit Euler.

The semi-implicit velocity update is formed using (from Stomakhin et al. [5])

$$\begin{aligned}
\boldsymbol{v}_{\boldsymbol{i}}^{n+1} &= \boldsymbol{v}_{\boldsymbol{i}}^n + \Delta t m_{\boldsymbol{i}}^{-1}((1-\beta)\boldsymbol{f}_{\boldsymbol{i}}^n + \beta \boldsymbol{f}_{\boldsymbol{i}}^{n+1}) \\
&= \boldsymbol{v}_{\boldsymbol{i}}^n + \Delta t m_{\boldsymbol{i}}^{-1}(\boldsymbol{f}_{\boldsymbol{i}}^n + \beta(\boldsymbol{f}_{\boldsymbol{i}}^{n+1} - \boldsymbol{f}_{\boldsymbol{i}}^n)) \\
&= (\boldsymbol{v}_{\boldsymbol{i}}^n + \Delta t m_{\boldsymbol{i}}^{-1}\boldsymbol{f}_{\boldsymbol{i}}^n) + \Delta t m_{\boldsymbol{i}}^{-1}\beta(\boldsymbol{f}_{\boldsymbol{i}}^{n+1} - \boldsymbol{f}_{\boldsymbol{i}}^n) \\
&= \boldsymbol{v}_{\boldsymbol{i}}^* + \Delta t m_{\boldsymbol{i}}^{-1}\beta\delta\boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}})
\end{aligned}$$

where $\beta \in [0, 1]$ for explicit vs semi-implicit integration.

This leads to a sparse system to solve for $\boldsymbol{v}_{\boldsymbol{i}}^{n+1}$ (where we can apply conjugate residual method)

$$\boldsymbol{v}_{\boldsymbol{i}}^{n+1} - \Delta t m_{\boldsymbol{i}}^{-1}\beta\delta\boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}}) = \boldsymbol{v}_{\boldsymbol{i}}^* \tag{36}$$

Notice that $\delta\boldsymbol{f}_{\boldsymbol{i}}$ relates to $\hat{\boldsymbol{x}}$, thus also dependent on $\boldsymbol{v}^{n+1}$.

The next few sections, whose derivations largely inspired by de Long [2], will give us $\delta\boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}})$.

6.2.1 Force derivative

The derivative of the grid node force is the following (for some arbitrary increment $\delta\boldsymbol{x}_{\boldsymbol{j}}$)

$$\delta\boldsymbol{f}_{\boldsymbol{i}}(\hat{\boldsymbol{x}}) = -\sum_{\boldsymbol{j}} \frac{\partial \boldsymbol{f}_{\boldsymbol{i}}}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}}(\hat{\boldsymbol{x}})\delta\boldsymbol{x}_{\boldsymbol{j}} \tag{37}$$

$$= -\sum_{\boldsymbol{j}} \frac{\partial^2 \Phi}{\partial \hat{\boldsymbol{x}}_{\boldsymbol{i}} \partial \hat{\boldsymbol{x}}_{\boldsymbol{j}}}(\hat{\boldsymbol{x}})\delta\boldsymbol{x}_{\boldsymbol{j}} \tag{38}$$

$$= -\sum_p V_p^0 \boldsymbol{A}_p (\boldsymbol{F}_{E_p}^n)^T \nabla w_{\boldsymbol{i}p}^n \tag{39}$$

where

$$\boldsymbol{A}_p = \frac{\partial^2 \Psi}{\partial \boldsymbol{F}_{E_p} \partial \boldsymbol{F}_{E_p}} (\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p}) : (\sum_{\boldsymbol{j}} \delta \boldsymbol{x}_{\boldsymbol{j}} (\nabla w^T_{\boldsymbol{j}p}) \boldsymbol{F}^n_{E_p}) \tag{40}$$

Recall that $\delta \boldsymbol{F}_{E_p}$ may be expressed as

$$\delta \boldsymbol{F}_{E_p} = \sum_{\boldsymbol{j}} \delta \boldsymbol{x}_{\boldsymbol{j}} (\nabla w^n_{\boldsymbol{j}p})^T \boldsymbol{F}^n_{E_p}$$

Let $\delta \boldsymbol{x}_{\boldsymbol{j}} = \Delta t \boldsymbol{v}^{n+1}_{\boldsymbol{j}}$ (the velocity that we wish to solve). Then the above is equivalent to

$$\delta \boldsymbol{F}_{E_p} = \sum_{\boldsymbol{j}} \Delta t \boldsymbol{v}^{n+1}_{\boldsymbol{j}} (\nabla w^n_{\boldsymbol{j}p})^T \boldsymbol{F}^n_{E_p} \tag{41}$$

$\boldsymbol{A}_p$ can then be rewritten as (from Stomakhin et al. [6])

$$\begin{aligned}
\boldsymbol{A}_p &= \frac{\partial^2 \Psi}{\partial \boldsymbol{F}_{E_p} \partial \boldsymbol{F}_{E_p}} (\hat{\boldsymbol{F}}_{E_p}(\hat{\boldsymbol{x}}), \boldsymbol{F}^n_{P_p}) : \delta \boldsymbol{F}_{E_p} & (42)\\
&= 2\mu(\delta \boldsymbol{F}_{E_p} - \delta \boldsymbol{R}_{E_p}) + \lambda J_{E_p} \boldsymbol{F}^{-T}_{E_p} (J_{E_p} \boldsymbol{F}^{-T}_{E_p} : \delta \boldsymbol{F}_{E_p}) + \lambda (J_{E_p} - 1) \delta (J_{E_p} \boldsymbol{F}^{-T}_{E_p}) & (43)
\end{aligned}$$

where

$$\begin{aligned}
\mu &= \mu_0 e^{\xi(1 - J_{P_p})}\\
\lambda &= \lambda_0 e^{\xi(1 - J_{P_p})}\\
J_{E_p} &= \det \boldsymbol{F}_{E_p}\\
J_{P_p} &= \det \boldsymbol{F}_{P_p}
\end{aligned}$$

The μ_0 and λ_0 are the initial Lamé parameters. The additional terms will be derived in the following sections. **The rest of this section will drop the $\Box_{E_p}$ subscript for ease of notation.**

Compute $\delta \boldsymbol{R}_{E_p}$

The technical report (from Stomakhin et al. [6]) suggests the following method to find $\delta \boldsymbol{R}_{E_p}$.

Recall $\boldsymbol{F} = \boldsymbol{RS}$ from polar decomposition, where $\boldsymbol{R}$ being a unitary matrix ($\boldsymbol{R}^* \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^* = \boldsymbol{I}$) and $\boldsymbol{S}$ being a positive-semidefinite Hermitian matrix ($\boldsymbol{S} = \boldsymbol{S}^*$).

$$\begin{aligned}
\delta \boldsymbol{F} &= \delta \boldsymbol{R} \boldsymbol{S} + \boldsymbol{R} \delta \boldsymbol{S} & (44)\\
\boldsymbol{R}^T \delta \boldsymbol{F} &= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} + \boldsymbol{S} & (\boldsymbol{R}^T \boldsymbol{R} = \boldsymbol{I})\\
\delta \boldsymbol{F}^T \boldsymbol{R} &= \boldsymbol{S}^T (\boldsymbol{R}^T \delta \boldsymbol{R})^T + \boldsymbol{S}^T &\\
\boldsymbol{R}^T \delta \boldsymbol{F} - \delta \boldsymbol{F}^T \boldsymbol{R} &= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} + \boldsymbol{S} - \boldsymbol{S}^T (\boldsymbol{R}^T \delta \boldsymbol{R})^T - \boldsymbol{S}^T &\\
&= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} + \boldsymbol{S} - \boldsymbol{S} (\boldsymbol{R}^T \delta \boldsymbol{R})^T - \boldsymbol{S} & (\boldsymbol{S} = \boldsymbol{S}^T)\\
&= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} - \boldsymbol{S} (\boldsymbol{R}^T \delta \boldsymbol{R})^T &
\end{aligned}$$

The technical report [6] showed that $\boldsymbol{R}^T \delta \boldsymbol{R}$ must be skew-symmetric. Then, with $(\boldsymbol{R}^T \delta \boldsymbol{R})^T = -\boldsymbol{R}^T \delta \boldsymbol{R}$, we have

$$\begin{aligned}
\boldsymbol{R}^T \delta \boldsymbol{F} - \delta \boldsymbol{F}^T \boldsymbol{R} &= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} - \boldsymbol{S} (\boldsymbol{R}^T \delta \boldsymbol{R})^T &\\
&= (\boldsymbol{R}^T \delta \boldsymbol{R}) \boldsymbol{S} + \boldsymbol{S} (\boldsymbol{R}^T \delta \boldsymbol{R}) & (45)
\end{aligned}$$

The equation above only tells us about $\boldsymbol{R}^T \delta \boldsymbol{R}$, but we really want $\delta \boldsymbol{R}$

$$
\begin{aligned}
\boldsymbol{R}^T\delta\boldsymbol{R} &= \boldsymbol{R}^T\delta\boldsymbol{R} \\
\boldsymbol{R}\boldsymbol{R}^T\delta\boldsymbol{R} &= \boldsymbol{R}(\boldsymbol{R}^T\delta\boldsymbol{R}) \\
\boldsymbol{I}\delta\boldsymbol{R} &= \boldsymbol{R}(\boldsymbol{R}^T\delta\boldsymbol{R}) && (\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I}) \\
\delta\boldsymbol{R} &= \boldsymbol{R}(\boldsymbol{R}^T\delta\boldsymbol{R}) && (46)
\end{aligned}
$$

Given $\boldsymbol{R}^T\delta\boldsymbol{R}$, the $\delta\boldsymbol{R}$ should not be difficult to compute. That leads us to compute $\boldsymbol{R}^T\delta\boldsymbol{R}$. Here we present a fairly exhaustively step-by-step derivation.

Again, since $\boldsymbol{R}^T\delta\boldsymbol{R}$ is skew-symmetric with diagonal entries of 0, we can write it as

$$
\boldsymbol{R}^T\delta\boldsymbol{R} = \begin{bmatrix} 0 & x & y \\ -x & 0 & z \\ -y & -z & 0 \end{bmatrix}
$$

$$
\begin{aligned}
&\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R} = (\boldsymbol{R}^T\delta\boldsymbol{R})\boldsymbol{S} + \boldsymbol{S}(\boldsymbol{R}^T\delta\boldsymbol{R}) \\
&= \begin{bmatrix} 0 & x & y \\ -x & 0 & z \\ -y & -z & 0 \end{bmatrix} \boldsymbol{S} + \boldsymbol{S} \begin{bmatrix} 0 & x & y \\ -x & 0 & z \\ -y & -z & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & x & y \\ -x & 0 & z \\ -y & -z & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{S}_{00} & \boldsymbol{S}_{10} & \boldsymbol{S}_{20} \\ \boldsymbol{S}_{01} & \boldsymbol{S}_{11} & \boldsymbol{S}_{21} \\ \boldsymbol{S}_{02} & \boldsymbol{S}_{12} & \boldsymbol{S}_{22} \end{bmatrix} + \begin{bmatrix} \boldsymbol{S}_{00} & \boldsymbol{S}_{10} & \boldsymbol{S}_{20} \\ \boldsymbol{S}_{01} & \boldsymbol{S}_{11} & \boldsymbol{S}_{21} \\ \boldsymbol{S}_{02} & \boldsymbol{S}_{12} & \boldsymbol{S}_{22} \end{bmatrix} \begin{bmatrix} 0 & x & y \\ -x & 0 & z \\ -y & -z & 0 \end{bmatrix} \\
&= \begin{bmatrix} x\boldsymbol{S}_{01} + y\boldsymbol{S}_{02} & x\boldsymbol{S}_{11} + y\boldsymbol{S}_{12} & x\boldsymbol{S}_{21} + y\boldsymbol{S}_{22} \\ -x\boldsymbol{S}_{00} + z\boldsymbol{S}_{02} & -x\boldsymbol{S}_{10} + z\boldsymbol{S}_{12} & -x\boldsymbol{S}_{20} + z\boldsymbol{S}_{22} \\ -y\boldsymbol{S}_{00} - z\boldsymbol{S}_{01} & -y\boldsymbol{S}_{10} - z\boldsymbol{S}_{11} & -y\boldsymbol{S}_{20} - z\boldsymbol{S}_{21} \end{bmatrix} + \begin{bmatrix} -x\boldsymbol{S}_{10} - y\boldsymbol{S}_{20} & x\boldsymbol{S}_{00} - z\boldsymbol{S}_{20} & y\boldsymbol{S}_{00} + z\boldsymbol{S}_{10} \\ -x\boldsymbol{S}_{11} - y\boldsymbol{S}_{21} & x\boldsymbol{S}_{01} - z\boldsymbol{S}_{21} & y\boldsymbol{S}_{01} + z\boldsymbol{S}_{11} \\ -x\boldsymbol{S}_{12} - y\boldsymbol{S}_{22} & x\boldsymbol{S}_{02} - z\boldsymbol{S}_{22} & y\boldsymbol{S}_{02} + z\boldsymbol{S}_{12} \end{bmatrix} \\
&= \begin{bmatrix} x(\boldsymbol{S}_{01} - \boldsymbol{S}_{10}) + y(\boldsymbol{S}_{02} - \boldsymbol{S}_{20}) & x(\boldsymbol{S}_{00} + \boldsymbol{S}_{11}) + y(\boldsymbol{S}_{12}) + z(-\boldsymbol{S}_{20}) & x(\boldsymbol{S}_{21}) + y(\boldsymbol{S}_{00} + \boldsymbol{S}_{22}) + z(\boldsymbol{S}_{10}) \\ x(-\boldsymbol{S}_{00} - \boldsymbol{S}_{11}) + y(-\boldsymbol{S}_{21}) + z(\boldsymbol{S}_{02}) & x(\boldsymbol{S}_{01} - \boldsymbol{S}_{10}) + z(\boldsymbol{S}_{12} - \boldsymbol{S}_{21}) & x(-\boldsymbol{S}_{20}) + y(\boldsymbol{S}_{01}) + z(\boldsymbol{S}_{11} + \boldsymbol{S}_{22}) \\ x(-\boldsymbol{S}_{12}) + y(-\boldsymbol{S}_{00} - \boldsymbol{S}_{22}) + z(-\boldsymbol{S}_{01}) & x(\boldsymbol{S}_{02}) + y(-\boldsymbol{S}_{10}) + z(-\boldsymbol{S}_{11} - \boldsymbol{S}_{22}) & y(\boldsymbol{S}_{02} - \boldsymbol{S}_{20}) + z(\boldsymbol{S}_{12} - \boldsymbol{S}_{21}) \end{bmatrix}
\end{aligned}
$$

Since $\boldsymbol{S}$ from polar decomposition is a positive-semidefinite Hermitian matrix ($\boldsymbol{S}_{ij} = \overline{\boldsymbol{S}_{ji}}$), it follows that

$$
\begin{aligned}
\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R} &= \begin{bmatrix} 0 & \xi_1 & \xi_2 \\ -\xi_1 & 0 & \xi_3 \\ -\xi_2 & -\xi_3 & 0 \end{bmatrix} \\
\xi_1 &= x(\boldsymbol{S}_{00} + \boldsymbol{S}_{11}) + y(\boldsymbol{S}_{12}) + z(-\boldsymbol{S}_{20}) = x(\boldsymbol{S}_{00} + \boldsymbol{S}_{11}) + y(\boldsymbol{S}_{21}) + z(-\boldsymbol{S}_{02}) \\
\xi_2 &= x(\boldsymbol{S}_{21}) + y(\boldsymbol{S}_{00} + \boldsymbol{S}_{22}) + z(\boldsymbol{S}_{10}) = x(\boldsymbol{S}_{12}) + y(\boldsymbol{S}_{00} + \boldsymbol{S}_{22}) + z(\boldsymbol{S}_{01}) \\
\xi_3 &= x(-\boldsymbol{S}_{20}) + y(\boldsymbol{S}_{01}) + z(\boldsymbol{S}_{11} + \boldsymbol{S}_{22}) = x(-\boldsymbol{S}_{02}) + y(\boldsymbol{S}_{10}) + z(\boldsymbol{S}_{11} + \boldsymbol{S}_{22})
\end{aligned}
$$

Then the entries of $\boldsymbol{R}^T\delta\boldsymbol{R}$ can be solved as

$$
\begin{aligned}
\begin{bmatrix} (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{10} \\ (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{20} \\ (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{21} \end{bmatrix} &= \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{S}_{00} + \boldsymbol{S}_{11} & \boldsymbol{S}_{12} & -\boldsymbol{S}_{20} \\ \boldsymbol{S}_{21} & \boldsymbol{S}_{00} + \boldsymbol{S}_{22} & \boldsymbol{S}_{10} \\ -\boldsymbol{S}_{20} & \boldsymbol{S}_{01} & \boldsymbol{S}_{11} + \boldsymbol{S}_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\
\begin{bmatrix} x \\ y \\ z \end{bmatrix} &= \begin{bmatrix} \boldsymbol{S}_{00} + \boldsymbol{S}_{11} & \boldsymbol{S}_{12} & -\boldsymbol{S}_{20} \\ \boldsymbol{S}_{21} & \boldsymbol{S}_{00} + \boldsymbol{S}_{22} & \boldsymbol{S}_{10} \\ -\boldsymbol{S}_{20} & \boldsymbol{S}_{01} & \boldsymbol{S}_{11} + \boldsymbol{S}_{22} \end{bmatrix}^{-1} \begin{bmatrix} (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{10} \\ (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{20} \\ (\boldsymbol{R}^T\delta\boldsymbol{F} - \delta\boldsymbol{F}^T\boldsymbol{R})_{21} \end{bmatrix} && (47)
\end{aligned}
$$

Then with x, y and z, we can compute $\delta\boldsymbol{R}$ with the equation derived earlier.

Compute $J_{E_p}\boldsymbol{F}_{E_p}^{-T}$

Since $J = \det \boldsymbol{F}$ and the inverse of a matrix is its cofactors transposed then divided by its determinant, we know the following (from de Long [2])

$$\begin{aligned} J\boldsymbol{F}^{-T} &= \text{cofactor}(\boldsymbol{F}) \qquad (48) \\ &= \begin{bmatrix} \boldsymbol{F}_{11}\boldsymbol{F}_{22} - \boldsymbol{F}_{12}\boldsymbol{F}_{21} & -\boldsymbol{F}_{01}\boldsymbol{F}_{22} + \boldsymbol{F}_{02}\boldsymbol{F}_{21} & \boldsymbol{F}_{01}\boldsymbol{F}_{12} - \boldsymbol{F}_{02}\boldsymbol{F}_{11} \\ -\boldsymbol{F}_{10}\boldsymbol{F}_{22} + \boldsymbol{F}_{12}\boldsymbol{F}_{20} & \boldsymbol{F}_{00}\boldsymbol{F}_{22} - \boldsymbol{F}_{02}\boldsymbol{F}_{20} & -\boldsymbol{F}_{00}\boldsymbol{F}_{12} + \boldsymbol{F}_{02}\boldsymbol{F}_{10} \\ \boldsymbol{F}_{10}\boldsymbol{F}_{21} - \boldsymbol{F}_{11}\boldsymbol{F}_{20} & -\boldsymbol{F}_{00}\boldsymbol{F}_{21} + \boldsymbol{F}_{01}\boldsymbol{F}_{20} & \boldsymbol{F}_{00}\boldsymbol{F}_{11} - \boldsymbol{F}_{01}\boldsymbol{F}_{10} \end{bmatrix} \qquad (49) \end{aligned}$$

Compute $\delta(J_{E_p}\boldsymbol{F}_{E_p}^{-T})$

The technical report (from Stomakhin et al. [6]) suggests the following

$$\delta(J\boldsymbol{F}^{-T}) = \frac{\partial}{\partial \boldsymbol{F}}(J\boldsymbol{F}^{-T}) : \delta\boldsymbol{F}$$

where the colon-notation here ($\boldsymbol{A} : \boldsymbol{B}$) represents taking a summation over the kl indices of $\boldsymbol{A}_{ijkl}\boldsymbol{B}_{kl}$.
Let's first watch the cofactor matrix of $\boldsymbol{F}$ (from Ding et al. [3])

$$\frac{\partial}{\partial \boldsymbol{F}}(J\boldsymbol{F}^{-T}) = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & \boldsymbol{F}_{22} & -\boldsymbol{F}_{12} \\ 0 & -\boldsymbol{F}_{21} & \boldsymbol{F}_{11} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ -\boldsymbol{F}_{22} & 0 & \boldsymbol{F}_{02} \\ \boldsymbol{F}_{21} & 0 & -\boldsymbol{F}_{01} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ \boldsymbol{F}_{12} & -\boldsymbol{F}_{02} & 0 \\ -\boldsymbol{F}_{11} & \boldsymbol{F}_{01} & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & -\boldsymbol{F}_{22} & \boldsymbol{F}_{12} \\ 0 & 0 & 0 \\ 0 & \boldsymbol{F}_{20} & -\boldsymbol{F}_{10} \end{bmatrix} & \begin{bmatrix} \boldsymbol{F}_{22} & 0 & -\boldsymbol{F}_{02} \\ 0 & 0 & 0 \\ -\boldsymbol{F}_{20} & 0 & \boldsymbol{F}_{00} \end{bmatrix} & \begin{bmatrix} -\boldsymbol{F}_{12} & \boldsymbol{F}_{02} & 0 \\ 0 & 0 & 0 \\ \boldsymbol{F}_{10} & -\boldsymbol{F}_{00} & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & \boldsymbol{F}_{21} & -\boldsymbol{F}_{11} \\ 0 & -\boldsymbol{F}_{20} & \boldsymbol{F}_{10} \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} -\boldsymbol{F}_{21} & 0 & \boldsymbol{F}_{01} \\ \boldsymbol{F}_{20} & 0 & -\boldsymbol{F}_{00} \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} \boldsymbol{F}_{11} & -\boldsymbol{F}_{01} & 0 \\ -\boldsymbol{F}_{01} & \boldsymbol{F}_{00} & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

Then we can compute each of the entries of $\delta(J\boldsymbol{F}^{-T})$ as

$$\begin{aligned} \delta(J\boldsymbol{F}^{-T})_{ij} &= \frac{\partial}{\partial \boldsymbol{F}}(J\boldsymbol{F}^{T})_{ij} : \delta\boldsymbol{F} \\ &= \sum_{kl} \frac{\partial}{\partial \boldsymbol{F}}(J\boldsymbol{F}^{T})_{ijkl}\delta\boldsymbol{F}_{kl} \end{aligned}$$

To live up to the exhaustiveness of this mostly complete handbook, we have

$$\delta(J\boldsymbol{F}^{-T}) = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & \boldsymbol{F}_{22} & -\boldsymbol{F}_{12} \\ 0 & -\boldsymbol{F}_{21} & \boldsymbol{F}_{11} \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} 0 & 0 & 0 \\ -\boldsymbol{F}_{22} & 0 & \boldsymbol{F}_{02} \\ \boldsymbol{F}_{21} & 0 & -\boldsymbol{F}_{01} \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} 0 & 0 & 0 \\ \boldsymbol{F}_{12} & -\boldsymbol{F}_{02} & 0 \\ -\boldsymbol{F}_{11} & \boldsymbol{F}_{01} & 0 \end{bmatrix} : \delta\boldsymbol{F} \\ \begin{bmatrix} 0 & -\boldsymbol{F}_{22} & \boldsymbol{F}_{12} \\ 0 & 0 & 0 \\ 0 & \boldsymbol{F}_{20} & -\boldsymbol{F}_{10} \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} \boldsymbol{F}_{22} & 0 & -\boldsymbol{F}_{02} \\ 0 & 0 & 0 \\ -\boldsymbol{F}_{20} & 0 & \boldsymbol{F}_{00} \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} -\boldsymbol{F}_{12} & \boldsymbol{F}_{02} & 0 \\ 0 & 0 & 0 \\ \boldsymbol{F}_{10} & -\boldsymbol{F}_{00} & 0 \end{bmatrix} : \delta\boldsymbol{F} \\ \begin{bmatrix} 0 & \boldsymbol{F}_{21} & -\boldsymbol{F}_{11} \\ 0 & -\boldsymbol{F}_{20} & \boldsymbol{F}_{10} \\ 0 & 0 & 0 \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} -\boldsymbol{F}_{21} & 0 & \boldsymbol{F}_{01} \\ \boldsymbol{F}_{20} & 0 & -\boldsymbol{F}_{00} \\ 0 & 0 & 0 \end{bmatrix} : \delta\boldsymbol{F} & \begin{bmatrix} \boldsymbol{F}_{11} & -\boldsymbol{F}_{01} & 0 \\ -\boldsymbol{F}_{01} & \boldsymbol{F}_{00} & 0 \\ 0 & 0 & 0 \end{bmatrix} : \delta\boldsymbol{F} \end{bmatrix} \qquad (50)$$

where the colon-notation here ($\boldsymbol{A} : \boldsymbol{B}$, a little overloaded) representa a Frobenius inner product (i.e. sum of element-wise products from both matrices).

References

[1] Thomas Breekveldt. Analysis of a material point method for snow. Master's thesis, 2017.

[2] Esther de Long. Simulating snow with the material point method. Master's thesis, Bournemouth University, 2015.

[3] Yang Ding, Luowen Qian, and Hai Zhang. Simulating snow in houdini, 2018.

[4] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, pages 24:1–24:52, New York, NY, USA, 2016. ACM.

[5] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Trans. Graph.*, 32(4):102:1–102:10, July 2013.

[6] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. Material point method for snow simulation. Technical report, 2013.