

# (Supplemental Material) MPM Time Step Breakdown for Hybrid Grains

YONGHAO YUE\*, The University of Tokyo  
BREANNAN SMITH\*, Columbia University  
PETER YICHEN CHEN\*, Columbia University  
MAYTEE CHANTHARAYUKHONTHORN\*, Massachusetts Institute of Technology  
KEN KAMRIN<sup>+</sup>, Massachusetts Institute of Technology  
EITAN GRINSPUN<sup>+</sup>, Columbia University

CCS Concepts: • **Computing methodologies** → **Physical simulation**;

Additional Key Words and Phrases: Granular materials, Material point method, Contact dynamics, Constraints, Physical simulation, MPM Pseudocode

## ACM Reference format:

Yonghao Yue\*, Breannan Smith\*, Peter Yichen Chen\*, Maytee Chantharayukhonthorn\*, Ken Kamrin<sup>+</sup>, and Eitan Grinspun<sup>+</sup>. 2018. (Supplemental Material) MPM Time Step Breakdown for Hybrid Grains. *ACM Trans. Graph.* 37, 6, Article 283 (November 2018), 2 pages.  
<https://doi.org/10.1145/3272127.3275095>

## 1 MPM TIME STEP BREAKDOWN

---

### Algorithm 1 MPM\_Step

---

1: MPM_Step_First_Phase	▷ Alg. 2
2: MPM_Step_Second_Phase	▷ Alg. 7

---

---

### Algorithm 2 MPM\_Step\_First\_Phase

---

1: Rasterize_Mass_And_Momentum_To_Grid	▷ Alg. 3
2: Compute_Stress_At_Points	▷ Alg. 4
3: Compute_Forces_On_Grid	▷ Alg. 5
4: Update_Momentum_On_Grid	▷ Alg. 6

---

---

### Algorithm 3 Rasterize\_Mass\_And\_Momentum\_To\_Grid

---

```
1: for point ∈ Material_Points do
2:   for node ∈ Stencil(point) do
3:      $w \leftarrow \text{Weight}(\text{point}, \text{node})$ 
4:      $\text{node}.m \ += \ w \cdot \text{point}.m$ 
5:      $\text{node}.p \ += \ w \cdot \text{point}.m \cdot \text{point}.v$ 
6:   end for
7: end for
```

---

\*Co-first authors – authors contributed equally.

<sup>+</sup>Corresponding authors (e-mail: [kkamrin@mit.edu](mailto:kkamrin@mit.edu), [eitan@cs.columbia.edu](mailto:eitan@cs.columbia.edu)).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

© 2018 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.

0730-0301/2018/11-ART283 \$15.00

<https://doi.org/10.1145/3272127.3275095>

---

### Algorithm 4 Compute\_Stress\_At\_Points

---

```
1: for point ∈ Material_Points do
2:    $\tau \leftarrow 0$ 
3:   if point. $J \leq 1$  then
4:      $\tau \leftarrow \frac{\kappa}{2} \cdot (\text{point}.J^2 - 1) \cdot I + \mu \cdot \text{dev}[\text{point}.\bar{b}^e]$ 
5:   end if
6:    $\text{point}.\sigma \leftarrow \tau / \text{point}.J$ 
7: end for
```

---

---

### Algorithm 5 Compute\_Forces\_On\_Grid

---

```
1: for point ∈ Material_Points do
2:   for node ∈ Stencil(point) do
3:      $\nabla w \leftarrow \text{Weight\_Grad}(\text{point}, \text{node})$ 
4:      $\text{node}.f \ += \ -\text{point}.V \cdot \text{point}.J \cdot \text{point}.\sigma \cdot \nabla w$ 
5:   end for
6: end for
7: for node ∈ Grid_Nodes do
8:    $\text{node}.f \ += \ \text{node}.m \cdot g$ 
9: end for
```

---

---

### Algorithm 6 Update\_Momentum\_On\_Grid

---

```
1: for node ∈ Grid_Nodes do
2:    $\text{node}.p_{new} \leftarrow \text{node}.p + dt \cdot \text{node}.f$ 
3: end for
```

---

---

### Algorithm 7 MPM\_Step\_Second\_Phase

---

1: Lumped_Mass_Velocity_Update_On_Grid	▷ Alg. 8
2: Compute_Velocity_Gradient_At_Points	▷ Alg. 9
3: Elastic_Prediction_At_Points	▷ Alg. 10
4: Plastic_Correction_At_Points	▷ Alg. 11
5: Update_Velocities_At_Points	▷ Alg. 12
6: Update_Positions_At_Points	▷ Alg. 13

---

---

### Algorithm 8 Lumped\_Mass\_Velocity\_Update\_On\_Grid

---

```
1: for node ∈ Grid_Nodes do
2:    $\text{node}.v \leftarrow \text{node}.p_{new} / \text{node}.m$ 
3:    $\text{node}.a \leftarrow (\text{node}.p_{new} - \text{node}.p) / (dt \cdot \text{node}.m)$ 
4: end for
```

---

**Algorithm 9** Compute\_Velocity\_Gradient\_At\_Points

---

```

1: for point  $\in$  Material_Points do
2:   point. $\nabla \mathbf{v} \leftarrow \mathbf{0}$ 
3:   for node  $\in$  Stencil(point) do
4:      $\nabla \mathbf{w} \leftarrow \text{Weight\_Grad}(\text{point}, \text{node})$ 
5:     point. $\nabla \mathbf{v} += \text{point}.\mathbf{v} \cdot \nabla \mathbf{w}^T$ 
6:   end for
7: end for

```

---

**Algorithm 10** Elastic\_Prediction\_At\_Points

---

```

1: for point  $\in$  Material_Points do
2:    $\mathbf{b}^e \leftarrow \text{point}.J \cdot \text{point}.\bar{\mathbf{b}}^e$ 
3:    $\mathbf{b}^{e*} \leftarrow \mathbf{b}^e + dt \cdot (\text{point}.\nabla \mathbf{v} \cdot \mathbf{b}^e + \mathbf{b}^e \cdot \text{point}.\nabla \mathbf{v}^T)$ 
4:   point. $J \leftarrow \sqrt{\det(\mathbf{b}^{e*})}$ 
5:   point. $\bar{\mathbf{b}}^e \leftarrow \mathbf{b}^{e*} / \text{point}.J$   $\triangleright$  if 2D
6:   point. $\bar{\mathbf{b}}^e \leftarrow \mathbf{b}^{e*} / (\text{point}.J)^{2/3}$   $\triangleright$  if 3D
7: end for

```

---

**Algorithm 11** Plastic\_Correction\_At\_Points

---

```

1: for point  $\in$  Material_Points do
2:   yield_threshold  $\leftarrow -\alpha \cdot \frac{\kappa}{2} \cdot (\text{point}.J^2 - 1)$ 
3:   dev[ $\mathbf{b}^e$ ]  $\leftarrow \text{point}.\mathbf{b}^e - \frac{1}{2} \cdot \text{Tr}[\text{point}.\mathbf{b}^e] \cdot \mathbf{I}$   $\triangleright$  if 2D
4:   dev[ $\mathbf{b}^e$ ]  $\leftarrow \text{point}.\mathbf{b}^e - \frac{1}{3} \cdot \text{Tr}[\text{point}.\mathbf{b}^e] \cdot \mathbf{I}$   $\triangleright$  if 3D
5:   dev[ $\bar{\mathbf{b}}^e$ ]  $\leftarrow \text{dev}[\mathbf{b}^e] / \text{point}.J$   $\triangleright$  if 2D
6:   dev[ $\bar{\mathbf{b}}^e$ ]  $\leftarrow \text{dev}[\mathbf{b}^e] / (\text{point}.J)^{2/3}$   $\triangleright$  if 3D
7:   if  $\mu \cdot \|\text{dev}[\bar{\mathbf{b}}^e]\|_F > \text{yield\_threshold}$  then
8:      $\lambda_2 \leftarrow \text{yield\_threshold} / (\mu \cdot \|\text{dev}[\bar{\mathbf{b}}^e]\|_F)$ 
9:      $\lambda_1 \leftarrow \sqrt{\det[\text{point}.\mathbf{b}^e] - \lambda_2^2 \cdot \det[\text{dev}[\mathbf{b}^e]]}$   $\triangleright$  if 2D
10:    Solve (17) for  $\lambda_1$  using Cardano's method  $\triangleright$  if 3D
11:    point. $\mathbf{b}^e \leftarrow \lambda_1 \cdot \mathbf{I} + \lambda_2 \cdot \text{dev}[\mathbf{b}^e]$ 
12:   end if
13: end for

```

---

**Algorithm 12** Update\_Velocities\_At\_Points

---

```

1: for point  $\in$  Material_Points do
2:    $\mathbf{v}_{pic} \leftarrow \mathbf{0}$ 
3:    $\mathbf{a}_{flip} \leftarrow \mathbf{0}$ 
4:   for node  $\in$  Stencil(point) do
5:      $w \leftarrow \text{Weight}(\text{point}, \text{node})$ 
6:      $\mathbf{v}_{pic} += w \cdot \text{node}.\mathbf{v}$ 
7:      $\mathbf{a}_{flip} += w \cdot \text{node}.\mathbf{a}$ 
8:   end for
9:    $\mathbf{v}_{flip} \leftarrow \text{point}.\mathbf{v} + dt \cdot \mathbf{a}_{flip}$ 
10:  point. $\mathbf{v} \leftarrow (1 - \beta) \cdot \mathbf{v}_{pic} + \beta \cdot \mathbf{v}_{flip}$ 
11: end for

```

---

**Algorithm 13** Update\_Positions\_At\_Points

---

```

1: for point  $\in$  Material_Points do
2:    $\mathbf{v}_{pic} \leftarrow \mathbf{0}$ 
3:   for node  $\in$  Stencil(point) do
4:      $w \leftarrow \text{Weight}(\text{point}, \text{node})$ 
5:      $\mathbf{v}_{pic} += w \cdot \text{node}.\mathbf{v}$ 
6:   end for
7:   point. $\mathbf{x} += dt \cdot \mathbf{v}_{pic}$ 
8: end for

```

---