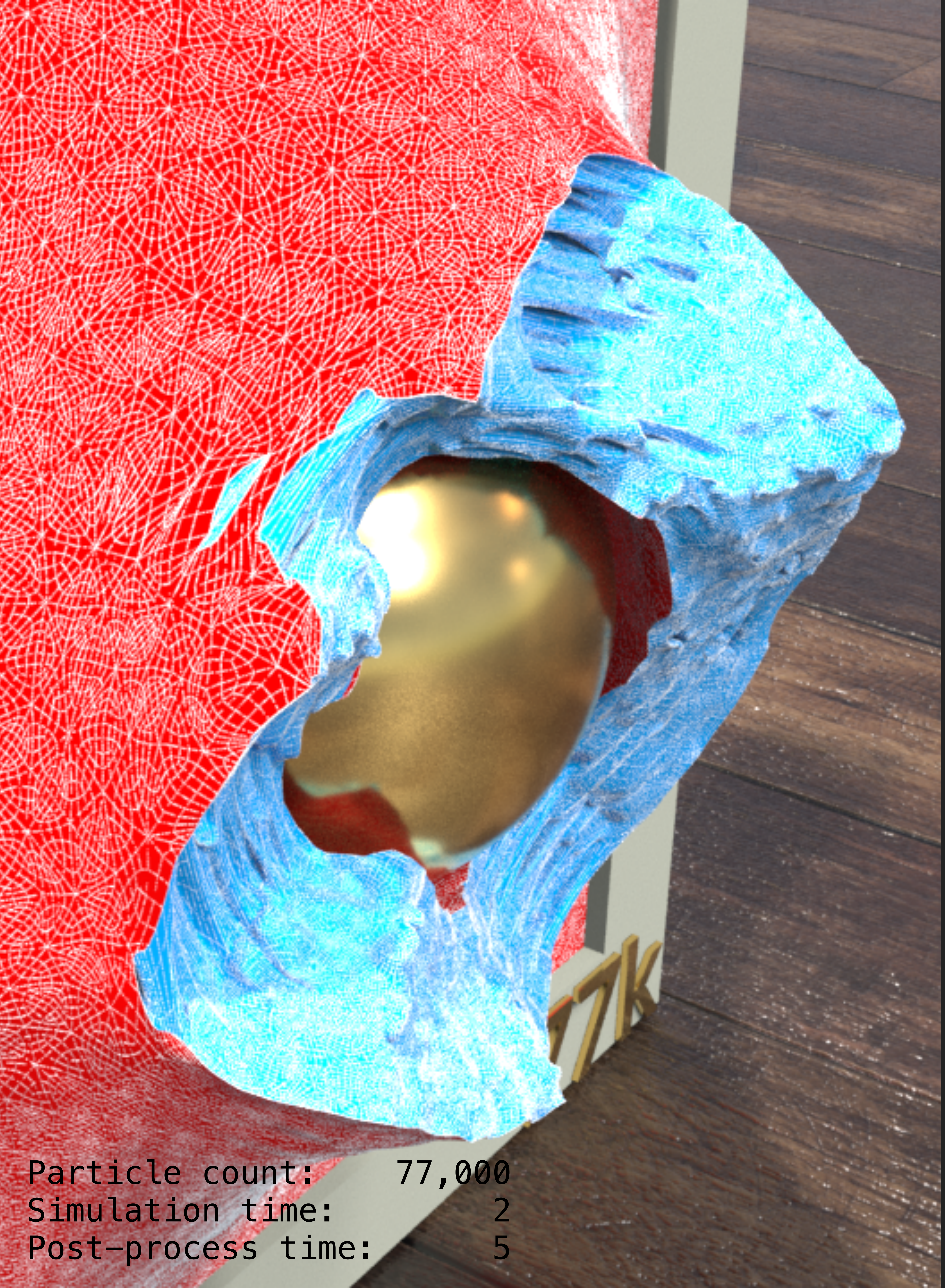


Stephanie Wang
University of California — Los Angeles
August 28, 2019

SIMULATION AND VISUALIZATION OF DUCTILE FRACTURE WITH MATERIAL POINT METHOD (MPM)

ADVISOR & COLLABORATORS

- Advisor: Joseph Teran, UCLA
- Mengyuan Ding, UCLA
- Theodore Gast, JIXIE EFFECTS (UCLA)
- Leyi Zhu, University of Science and Technology of China
- Steven Gagniere, UCLA
- Chenfanfu Jiang, University of Pennsylvania (UCLA)


Particle count: 60,000
Simulation time: 11
Post-process time: N/A


Particle count: 60,000
Simulation time: 11
Post-process time: 5


Particle count: 207,000
Simulation time: 16
Post-process time: 13

OUTLINE

- Material Point Method (MPM)
  - Grid-particle transfer
  - Force computation
- Simulation and visualization of ductile fracture
  - Yield surfaces
  - Mesh-processing
- Discussion

THE MATERIAL POINT METHOD

ROUGH ALGORITHM

- Particles for state
- Grid for computations
- Similar to FEM:
  - Vertices for state
  - Mesh for computations
- Interpolation between particles and grid

ROUGH ALGORITHM

$$m_i^n = \textsc{TransferP2G}(m_p)$$
$$\mathbf{v}_i^n = \textsc{TransferP2G}(\mathbf{v}_p^n)$$
$$\mathbf{f}_i^n = \textsc{ComputeForce}()$$
$$\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \frac{\Delta t}{m_i^n}\mathbf{f}_i^n$$
$$\mathbf{v}_p^{n+1} = \textsc{TransferG2P}(\tilde{\mathbf{v}}_i^{n+1})$$
$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$$

notation	meaning	when	where
$\mathbf{x}_p^{n+1}$	position	after forces	particle
$\mathbf{v}_i^n$	velocity	before forces	grid
m_p	mass	never changes	particle

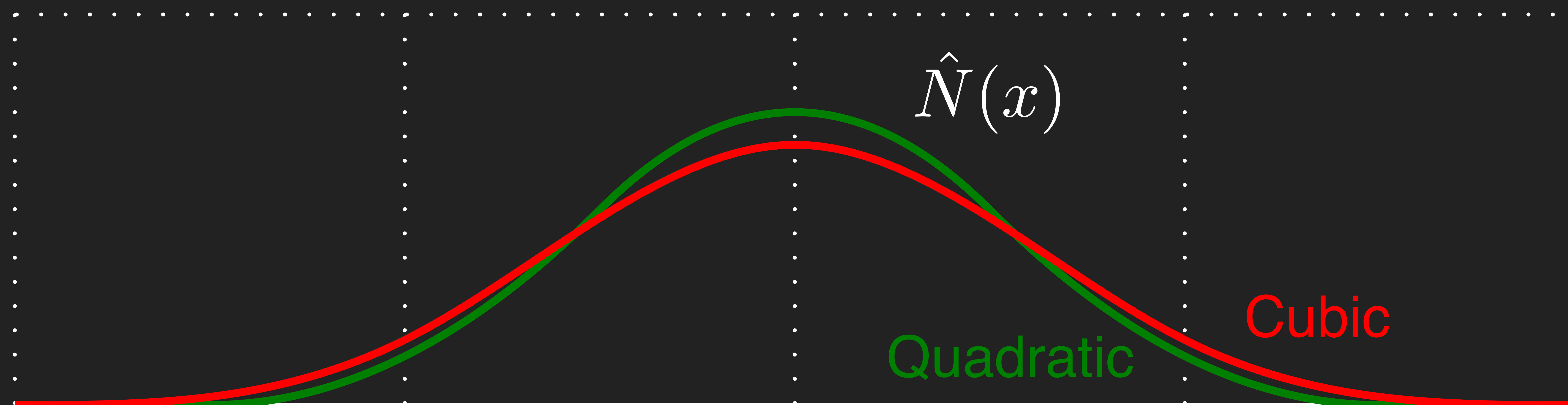
ROUGH ALGORITHM

$$m_i^n = \textsc{TransferP2G}(m_p)$$
$$\mathbf{v}_i^n = \textsc{TransferP2G}(\mathbf{v}_p^n)$$
$$\mathbf{f}_i^n = \textsc{ComputeForce}()$$
$$\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \frac{\Delta t}{m_i^n}\mathbf{f}_i^n$$
$$\mathbf{v}_p^{n+1} = \textsc{TransferG2P}(\tilde{\mathbf{v}}_i^{n+1})$$
$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$$

notation	meaning	when	where
$\mathbf{x}_p^{n+1}$	position	after forces	particle
$\mathbf{v}_i^n$	velocity	before forces	grid
m_p	mass	never changes	particle

INTERPOLATION SCHEME

- Compactly supported kernel function
- Spline: C1 (C2) piecewise-polynomial



INTERPOLATION SCHEME

- Tensor product: $N(\mathbf{x}) = \hat{N}(x)\hat{N}(y)\hat{N}(z)$
- Compute weights: $w_{ip}^n = N(\mathbf{x}_i^n - \mathbf{x}_p^n)$

  $\nabla w_{ip}^n = \nabla N(\mathbf{x}_i^n - \mathbf{x}_p^n)$
- Partition of unity $\sum_i w_{ip}^n = 1$
- Barycentric embedding $\sum_i w_{ip}^n \mathbf{x}_i^n = \mathbf{x}_p^n$
- Conservation of momenta, non-increasing energy

INTERPOLATION SCHEME

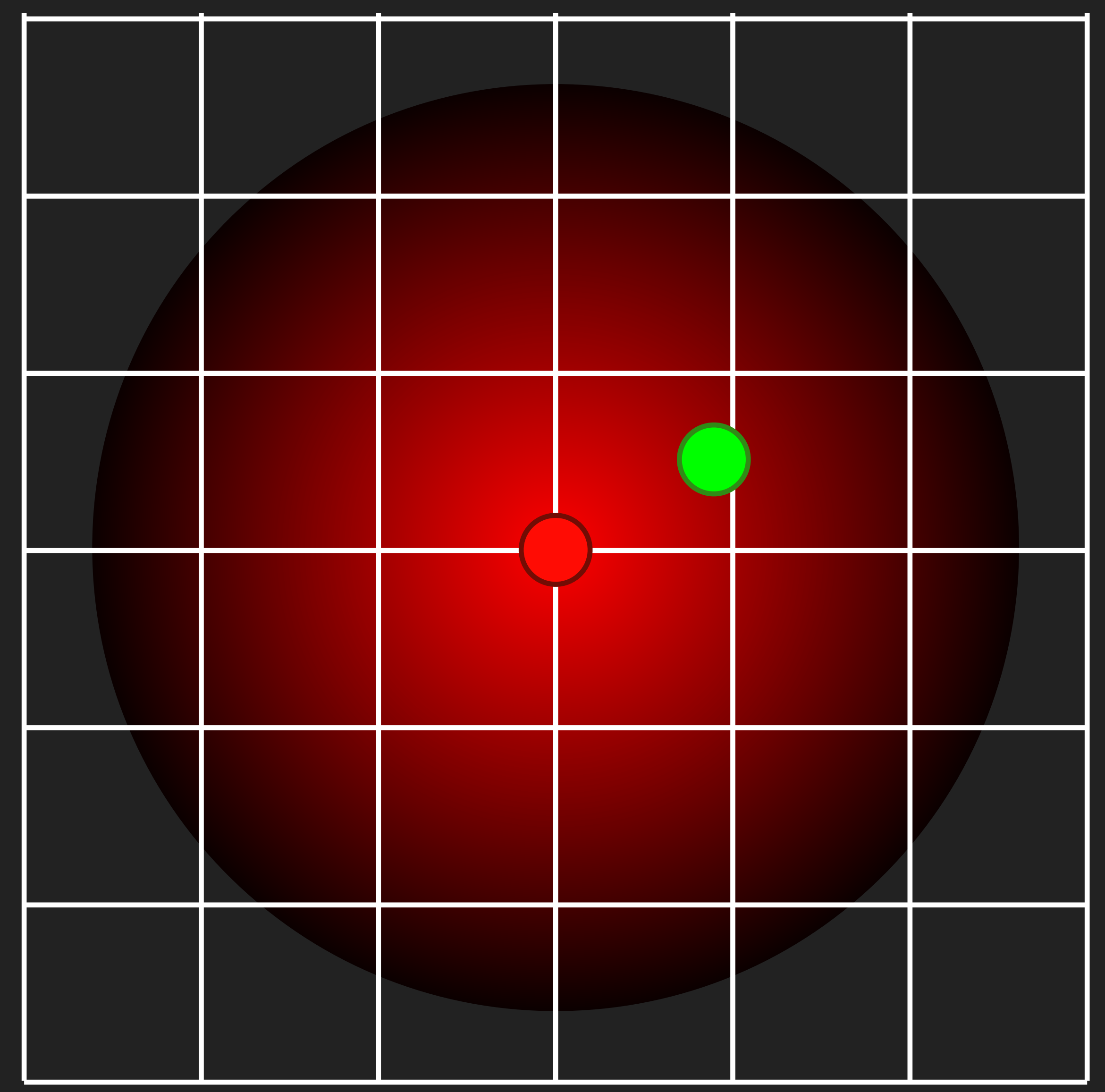
TransferP2G

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

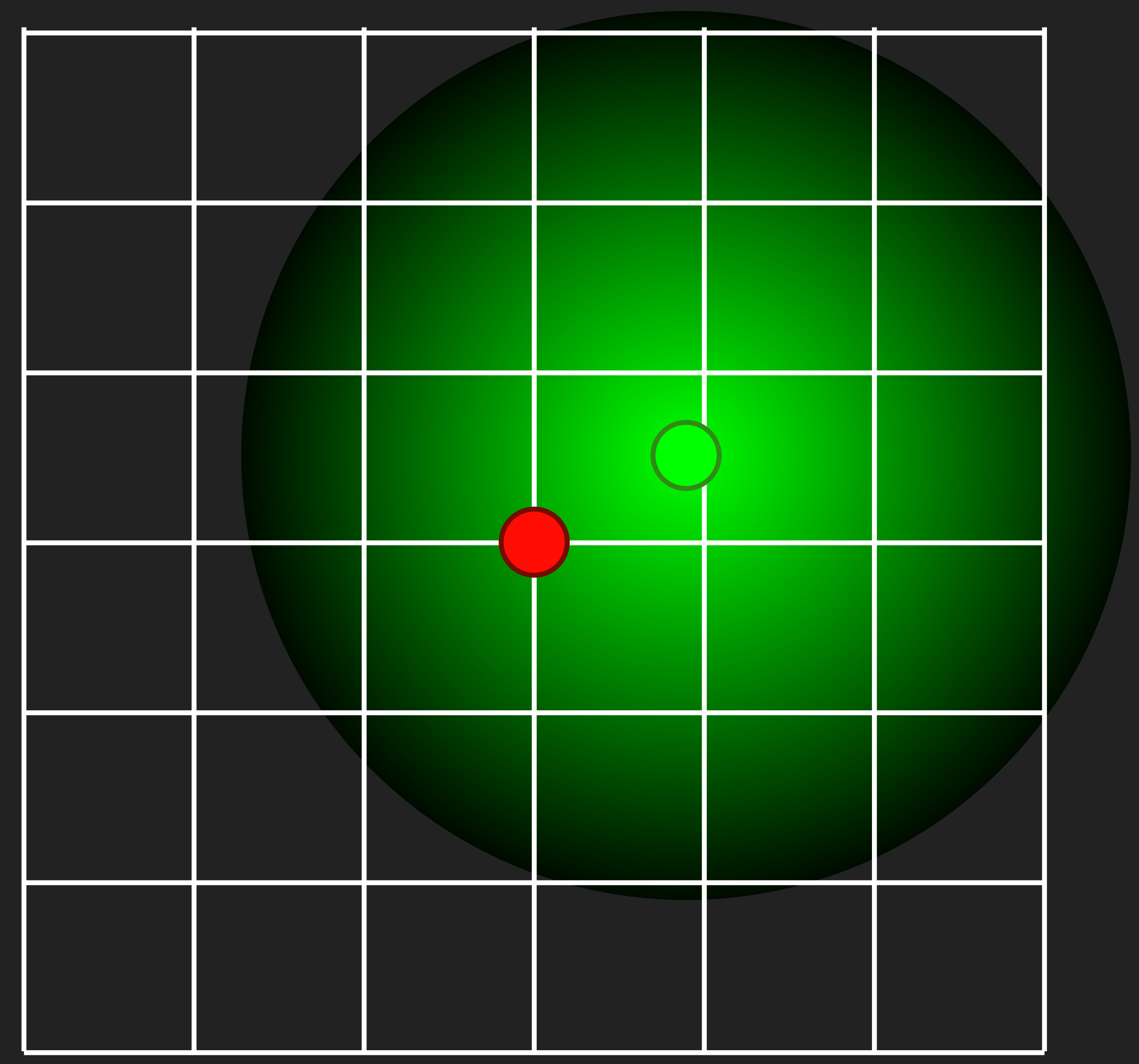
$$m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p \mathbf{v}_p^n$$ Momentum

TransferG2P

$$\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}$$



Kernel at node



Kernel at particle

PIC, FLIP, APIC, RPIC, ...

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

$$\mathbf{v}_i^n = \frac{1}{m_i^n} \sum_p w_{ip}^n m_p \mathbf{v}_p^n$$

$$\mathbf{f}_i^n = \textsc{ComputeForce}()$$

$$\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \frac{\Delta t}{m_i^n} \mathbf{f}_i^n$$

$$\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}$$

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

Particle In Cell (PIC)

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

$$\mathbf{D}_p^n = \sum_i w_{ip}^n (\mathbf{x}_i^n - \mathbf{x}_p^n)(\mathbf{x}_i^n - \mathbf{x}_p^n)^T$$

$$\mathbf{v}_i^n = \frac{1}{m_i^n} \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1} (\mathbf{x}_i^n - \mathbf{x}_p^n))$$

$$\mathbf{f}_i^n = \textsc{ComputeForce}()$$

$$\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \frac{\Delta t}{m_i^n} \mathbf{f}_i^n$$

$$\mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}$$

$$\mathbf{B}_p^{n+1} = \sum_i w_{ip}^n \mathbf{v}_i^n (\mathbf{x}_i^n - \mathbf{x}_p^n)^T$$

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

Affine Particle In Cell (APIC)

PIC, FLIP, APIC, RPIC, …….

- Particle In Cell (PIC): Harlow 1964
- Fluid Implicit Particle (FLIP): Brackbill and Ruppel 1986
- Affine Particle In Cell (APIC): Jiang et al. 2015
- Rigid Particle In Cell (RPIC): Jiang et al. 2015
- Polynomial Particle In Cell (PolyPIC): Fu et al. 2017
- Extended Particle In Cell (XPIC): Hammerquist et al. 2017

ROUGH ALGORITHM

$$
\begin{aligned}
m_i^n &= \textsc{TransferP2G}(m_p) \\
\mathbf{v}_i^n &= \textsc{TransferP2G}(\mathbf{v}_p^n) \\
\mathbf{f}_i^n &= \textsc{ComputeForce}() \\
\tilde{\mathbf{v}}_i^{n+1} &= \mathbf{v}_i^n + \frac{\Delta t}{m_i^n}\mathbf{f}_i^n \\
\mathbf{v}_p^{n+1} &= \textsc{TransferG2P}(\tilde{\mathbf{v}}_i^{n+1}) \\
\mathbf{x}_p^{n+1} &= \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}
\end{aligned}
$$

notation	meaning	when	where
$\mathbf{x}_p^{n+1}$	velocity	before forces	grid
$\mathbf{v}_i^n$	position	after forces	particle
m_p	mass	never changes	particle

ROUGH ALGORITHM

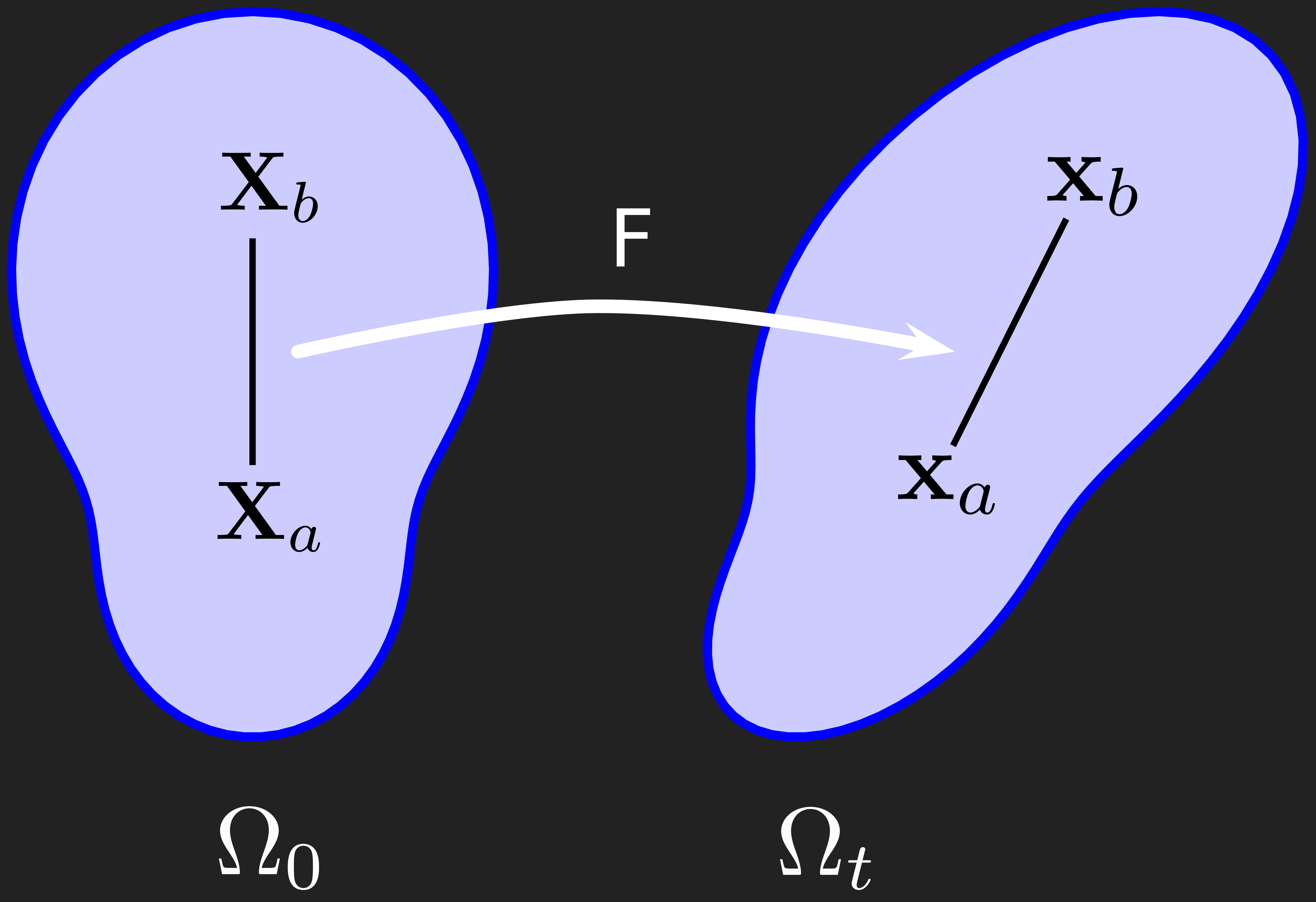
$$m_i^n = \textsc{TransferP2G}(m_p)$$
$$\mathbf{v}_i^n = \textsc{TransferP2G}(\mathbf{v}_p^n)$$
$$\boxed{\mathbf{f}_i^n = \textsc{ComputeForce}()}$$
$$\tilde{\mathbf{v}}_i^{n+1} = \mathbf{v}_i^n + \frac{\Delta t}{m_i^n}\mathbf{f}_i^n$$
$$\mathbf{v}_p^{n+1} = \textsc{TransferG2P}(\tilde{\mathbf{v}}_i^{n+1})$$
$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$$

notation	meaning	when	where
$\mathbf{x}_p^{n+1}$	velocity	before forces	grid
$\mathbf{v}_i^n$	position	after forces	particle
m_p	mass	never changes	particle

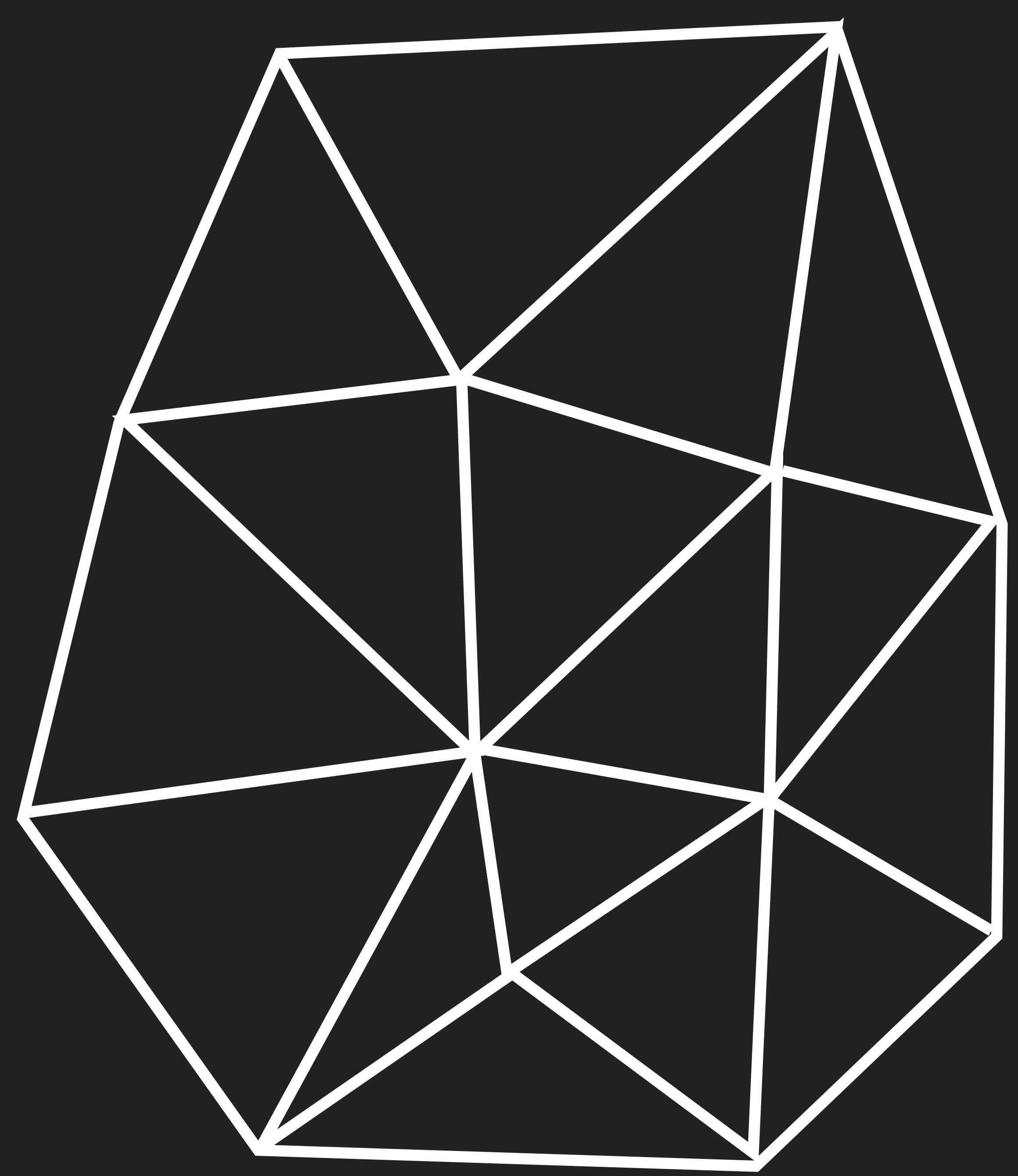
DEFORMATION GRADIENT

$$\mathbf{x} = \Phi(\mathbf{X}, t)$$

$$\mathbf{F}(\mathbf{X}, t) = \frac{\partial \Phi}{\partial \mathbf{X}}(\mathbf{X}, t)$$

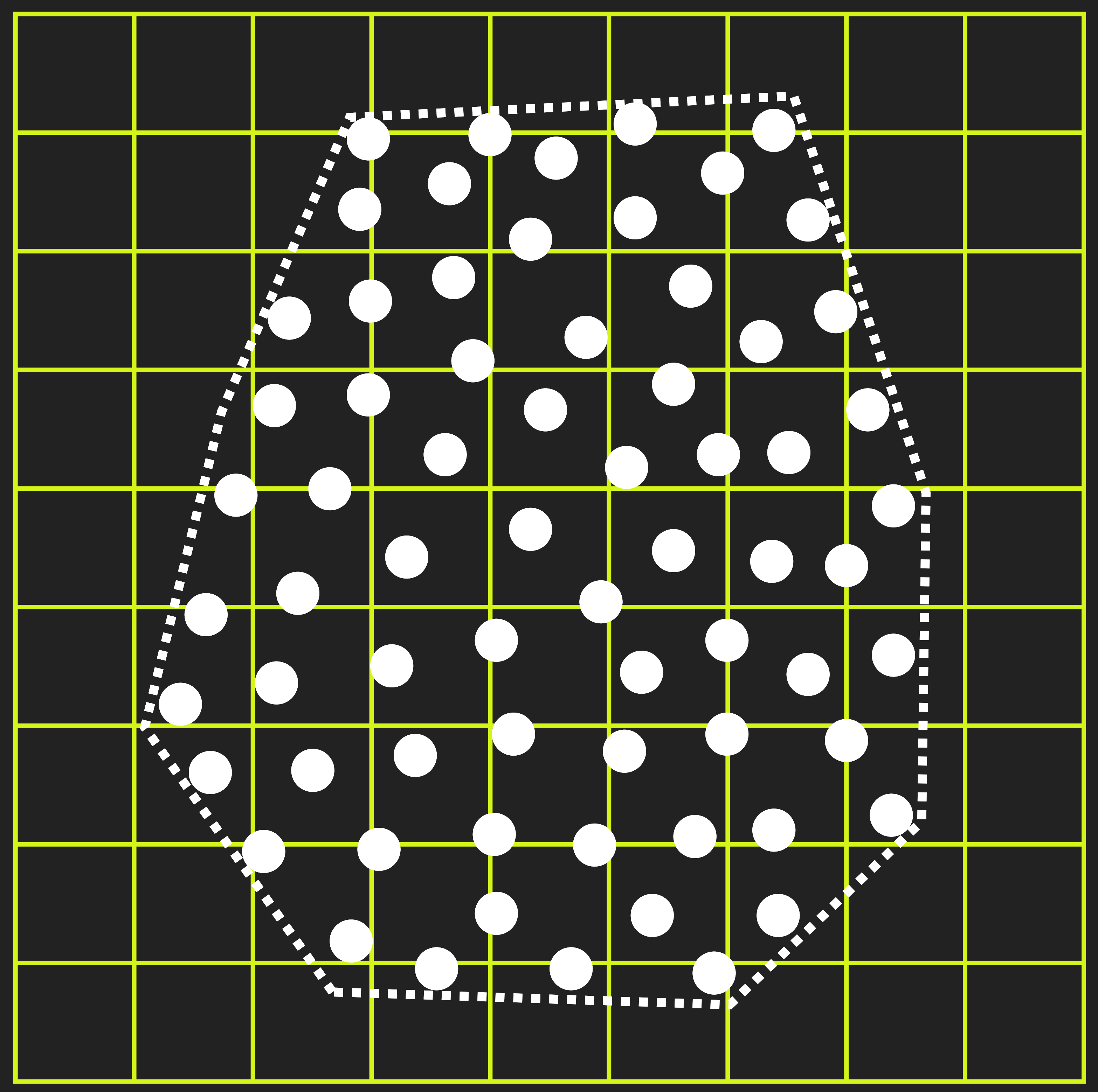


DEFORMATION GRADIENT



mesh-based forces:
F per triangle

$$\Phi = \sum_e V_e^0 \Psi(\mathbf{F}_e)$$



particle-based forces:
F per particle

$$\Phi = \sum_p V_p^0 \Psi(\mathbf{F}_p)$$

FORCE AS ENERGY GRADIENT

- First Piola-Kirchoff stress $\mathbf{P}(\mathbf{F}) = \frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{F})$
- Total potential energy $\Phi = \sum_p V_p^0 \Psi(\mathbf{F}_p)$
  - "F is a function of x" $\mathbf{F}_p^{n+1} = \left(\mathbf{I} + \Delta t \sum_i \mathbf{v}_i (\nabla \omega_{ip}^n)^T\right) \mathbf{F}_p^n$ $\quad \mathbf{F}_e^n = \sum_q \mathbf{x}_q^n \nabla N_q(\mathbf{X}_e)^T$
  - Energy is a function of x $\mathbf{f}_i = -\frac{\partial \Phi}{\partial \mathbf{x}_i}$
  - Force can be computed from x

$$\mathbf{f}_i = -\frac{\partial \Phi}{\partial \mathbf{x}_i} = -\sum_p V_p^0 \left(\frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{F}_p(\mathbf{x}))\right) (\mathbf{F}_p^n)^T \nabla \omega_{ip}^n$$

HYPER-ELASTIC MODELS

- St. Venant Kirchhoff potential with Hencky strain

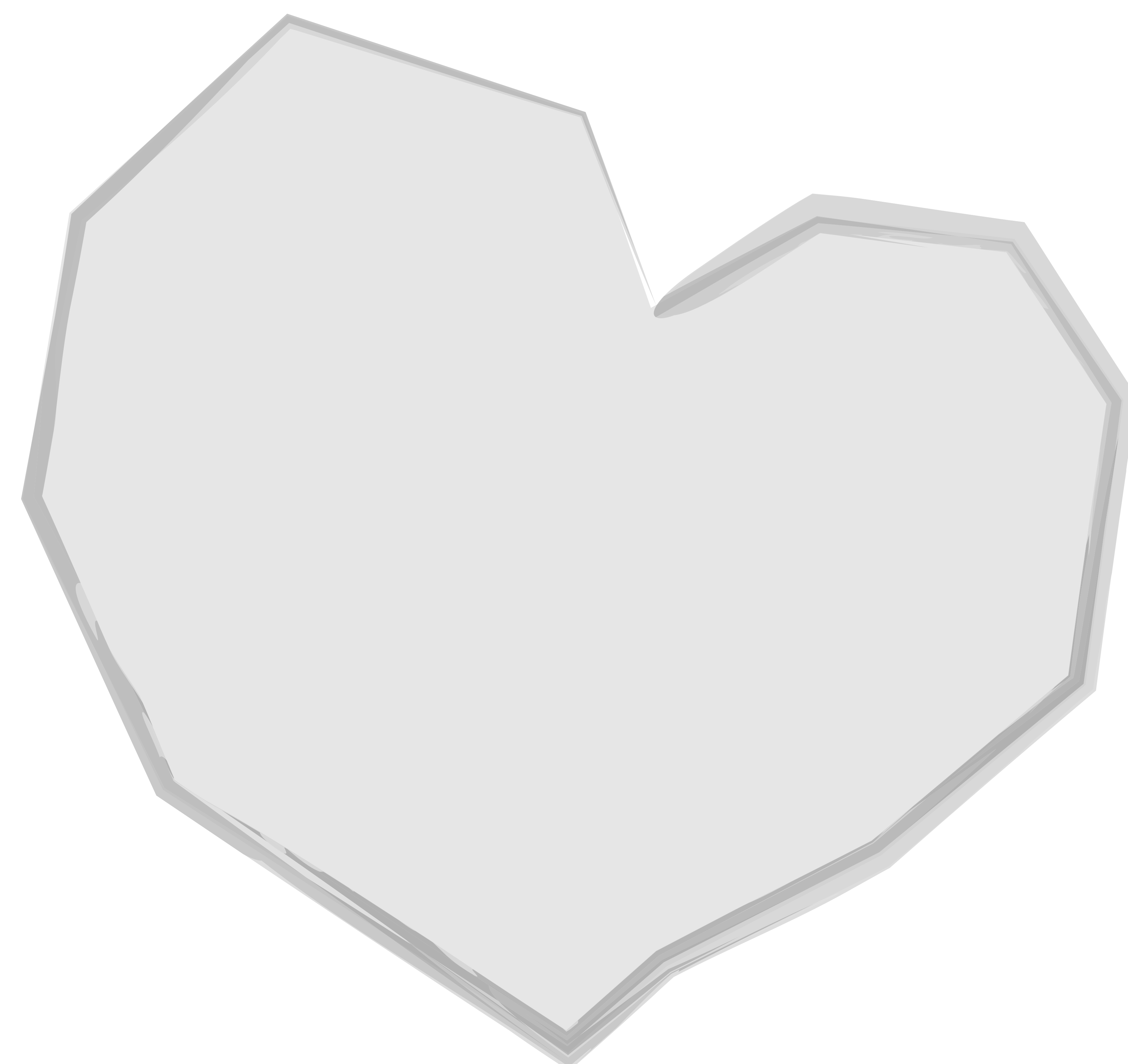
$$\mathbf{F} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

$$\psi(\mathbf{F}) = \mu \mathrm{tr}((\ln \boldsymbol{\Sigma})^2) + \frac{\lambda}{2}(\mathrm{tr}(\ln \boldsymbol{\Sigma}))^2$$

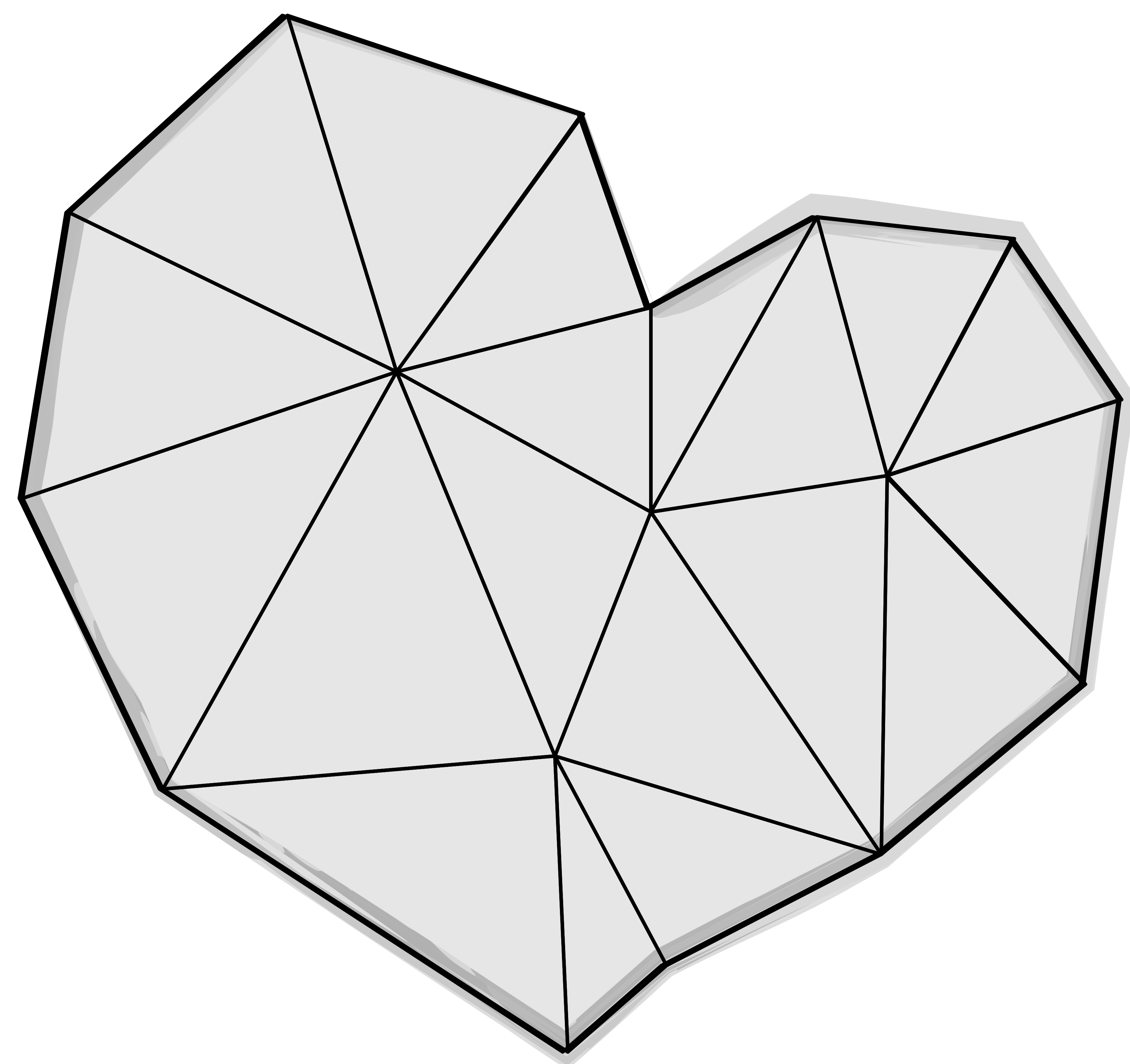
$$\frac{\partial \psi}{\partial \mathbf{F}} = \mathbf{U}(2\mu\boldsymbol{\Sigma}^{-1}\ln \boldsymbol{\Sigma} + \lambda \mathrm{tr}(\ln \boldsymbol{\Sigma})\boldsymbol{\Sigma}^{-1})\mathbf{V}^T$$

- (Easy for analytical plastic projection)

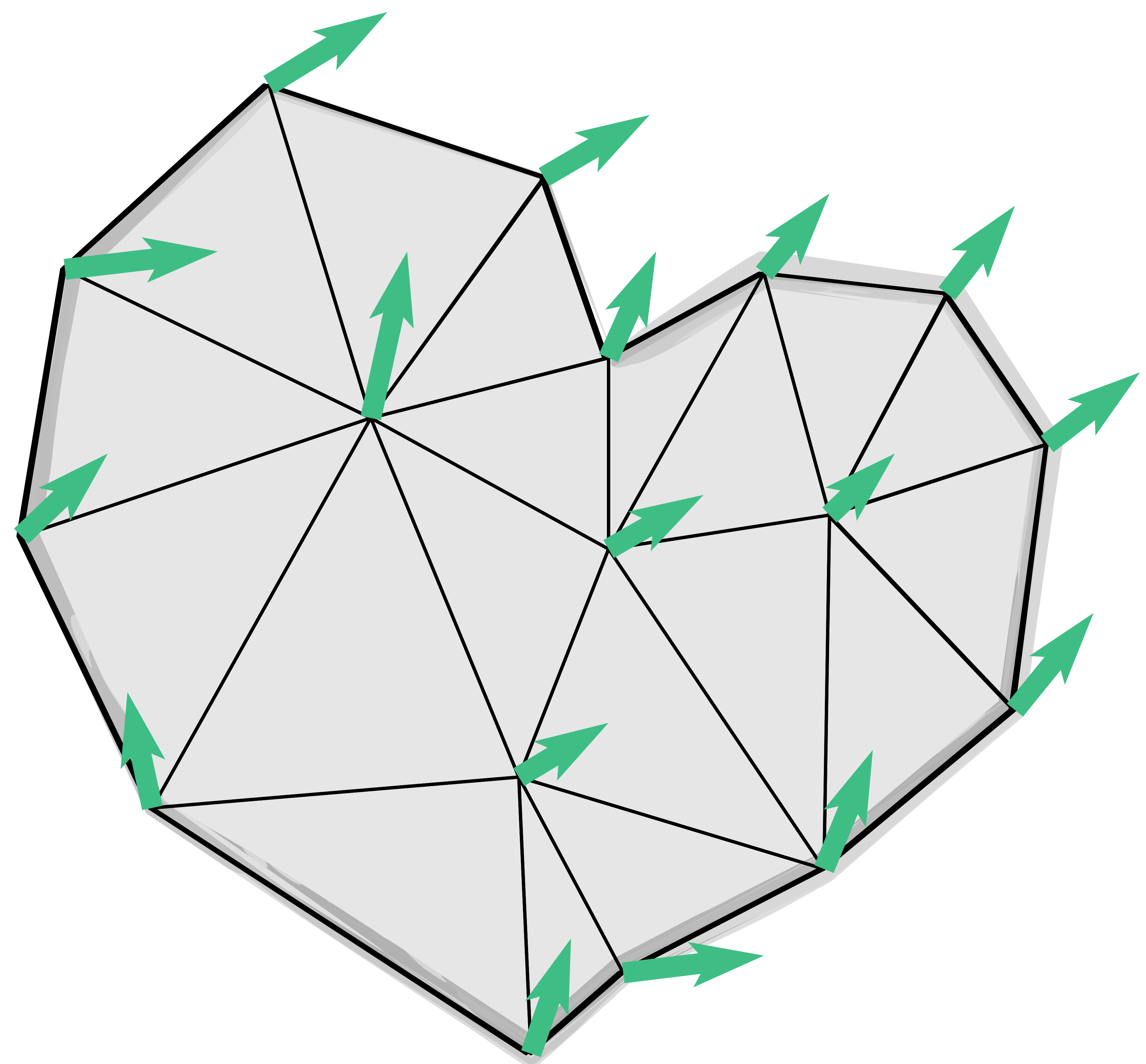
FINITE ELEMENT ELEMENT



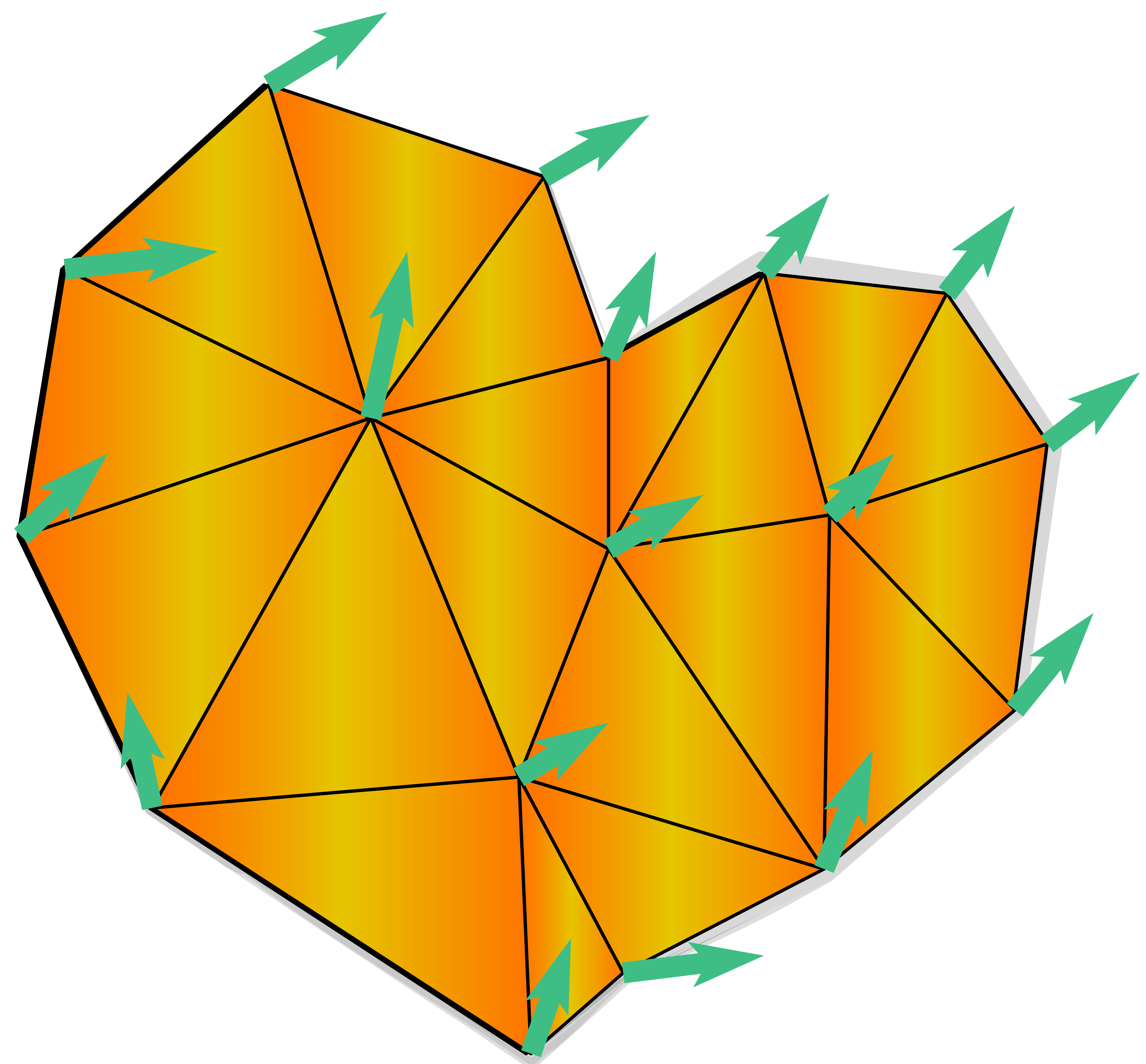
FINITE ELEMENT ELEMENT



FINITE ELEMENT ELEMENT



FINITE ELEMENT ELEMENT

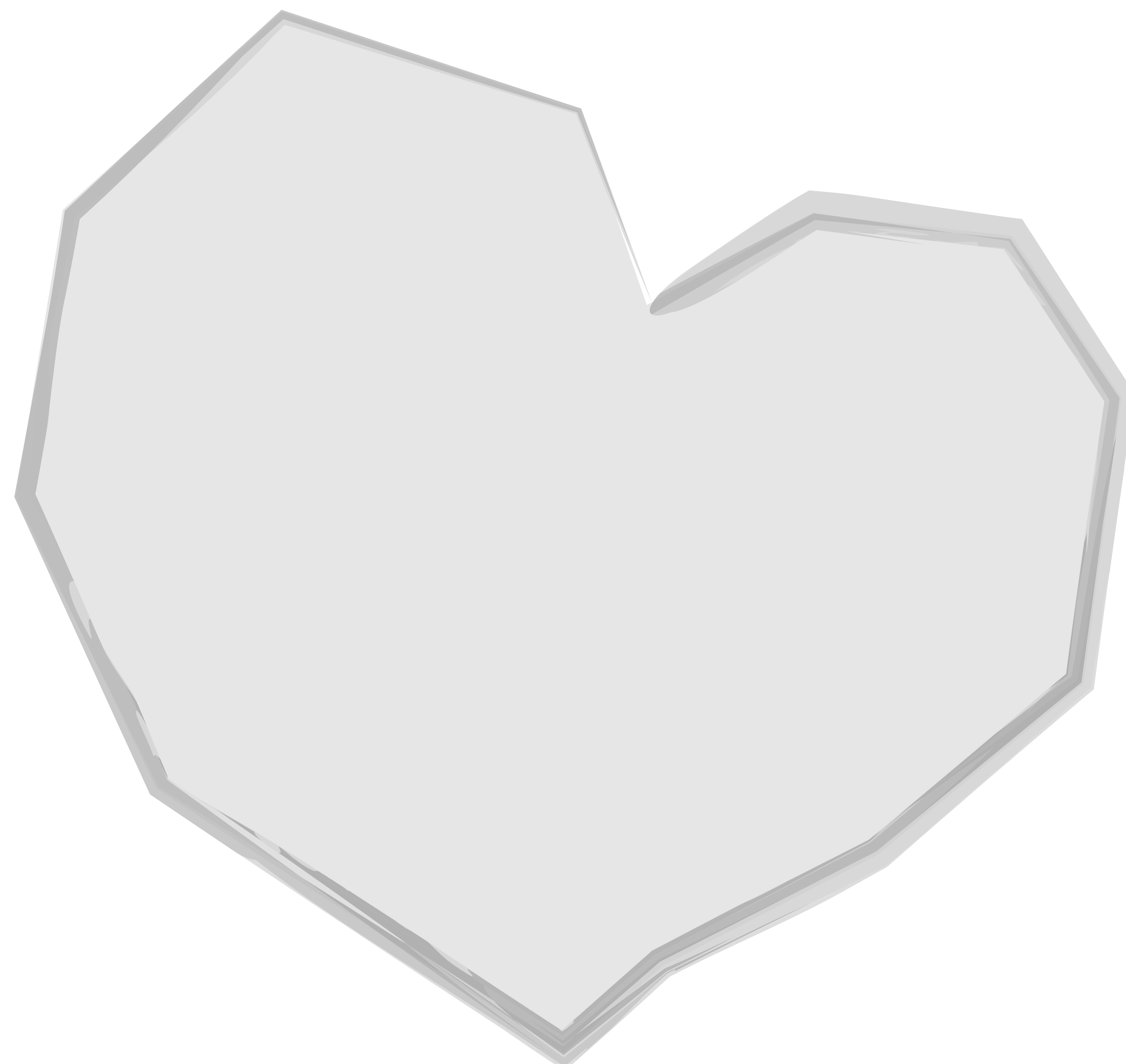


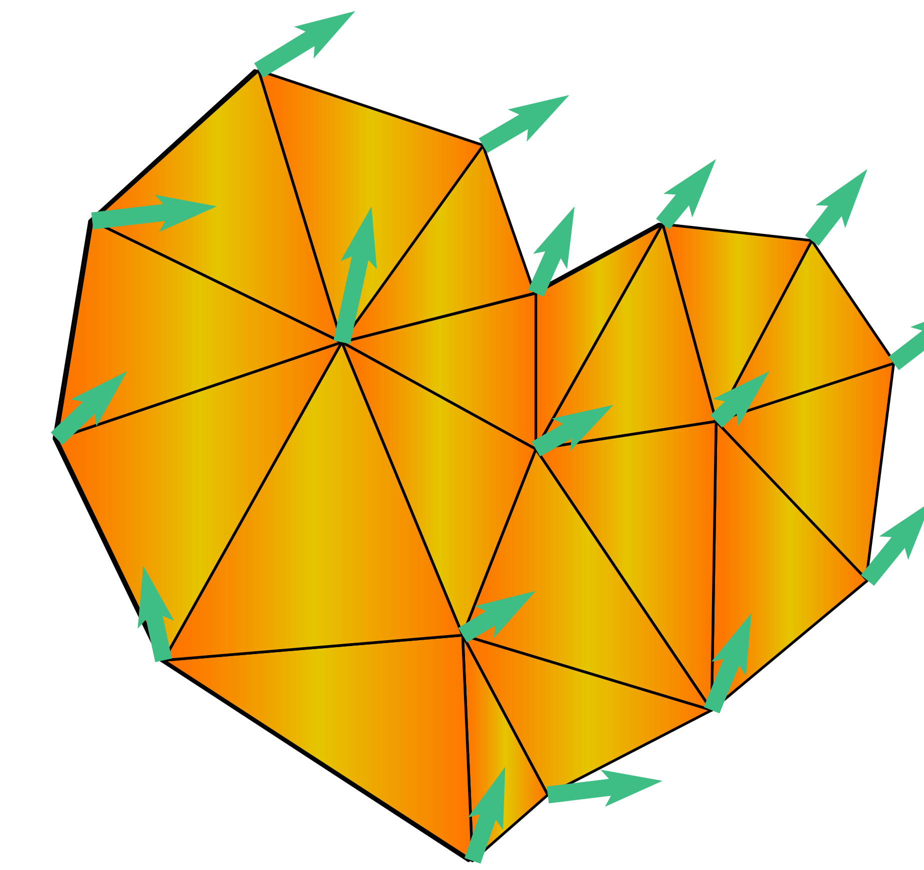
$$\Phi = \sum_e V_e^0 \Psi(\mathbf{F}_e)$$

$$\mathbf{F}_e^n = \sum_q \mathbf{x}_q^n \nabla N_q(\mathbf{X}_e)^T$$

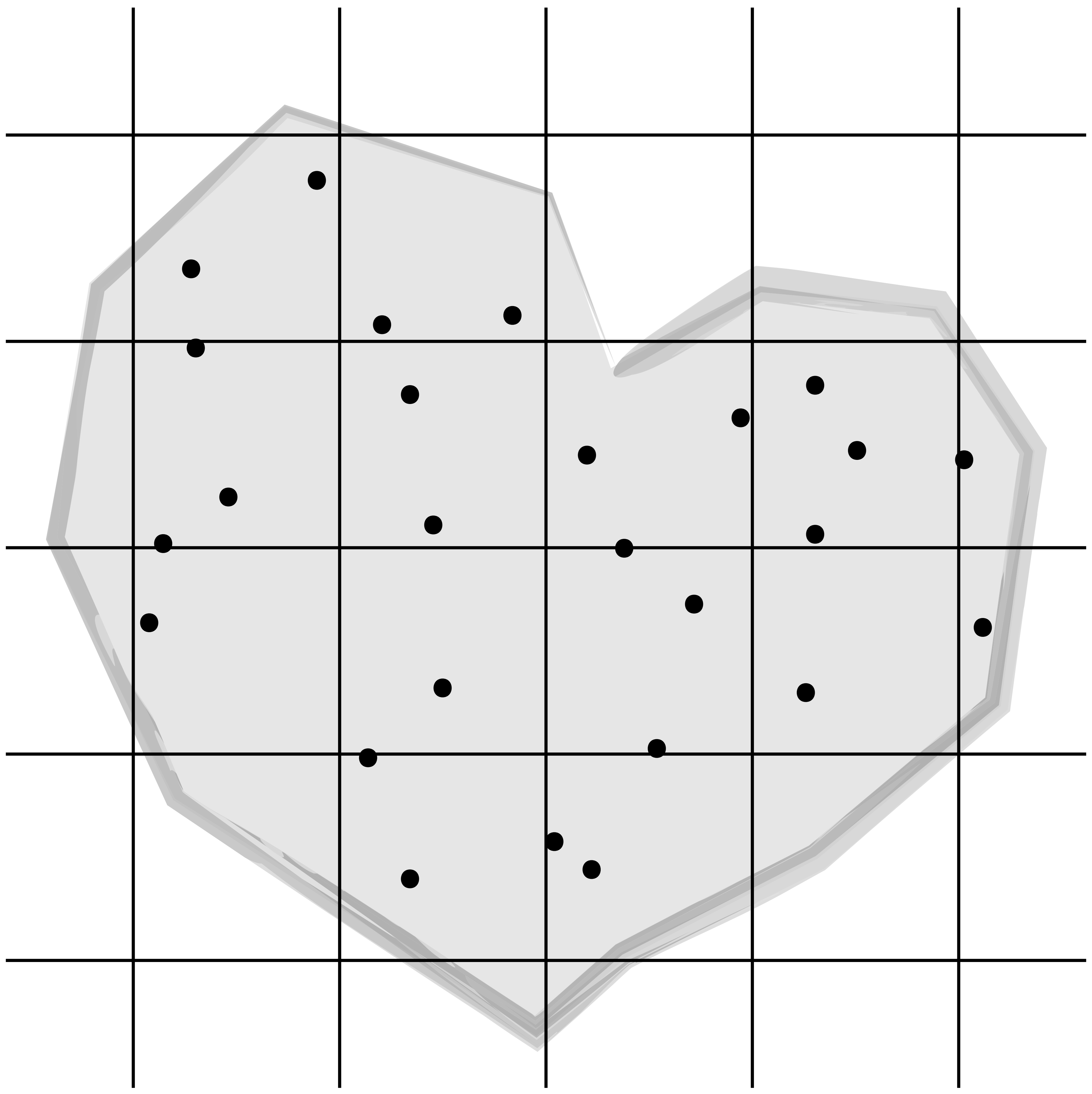
$$\mathbf{F}_e^n = \left(\sum_q \mathbf{x}_q^n \nabla N_q(\xi_e)^T\right)\left(\sum_q \mathbf{X}_q \nabla N_q(\xi_e)^T\right)^{-1}$$

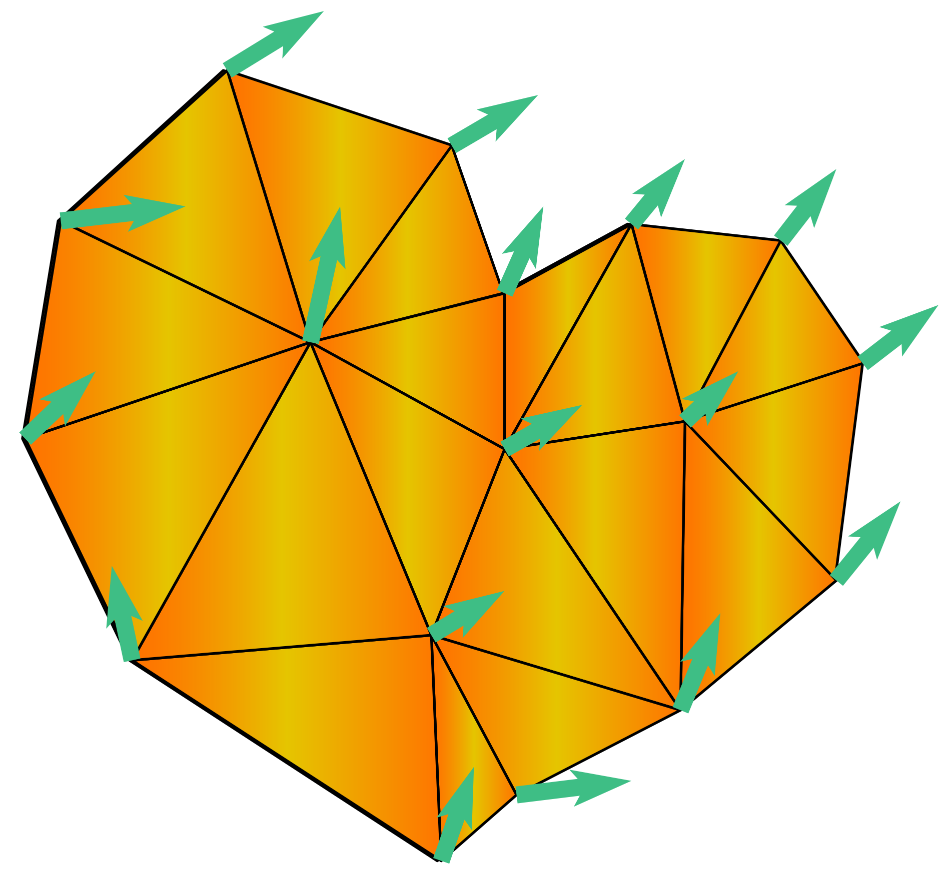
PARTICLE MPM



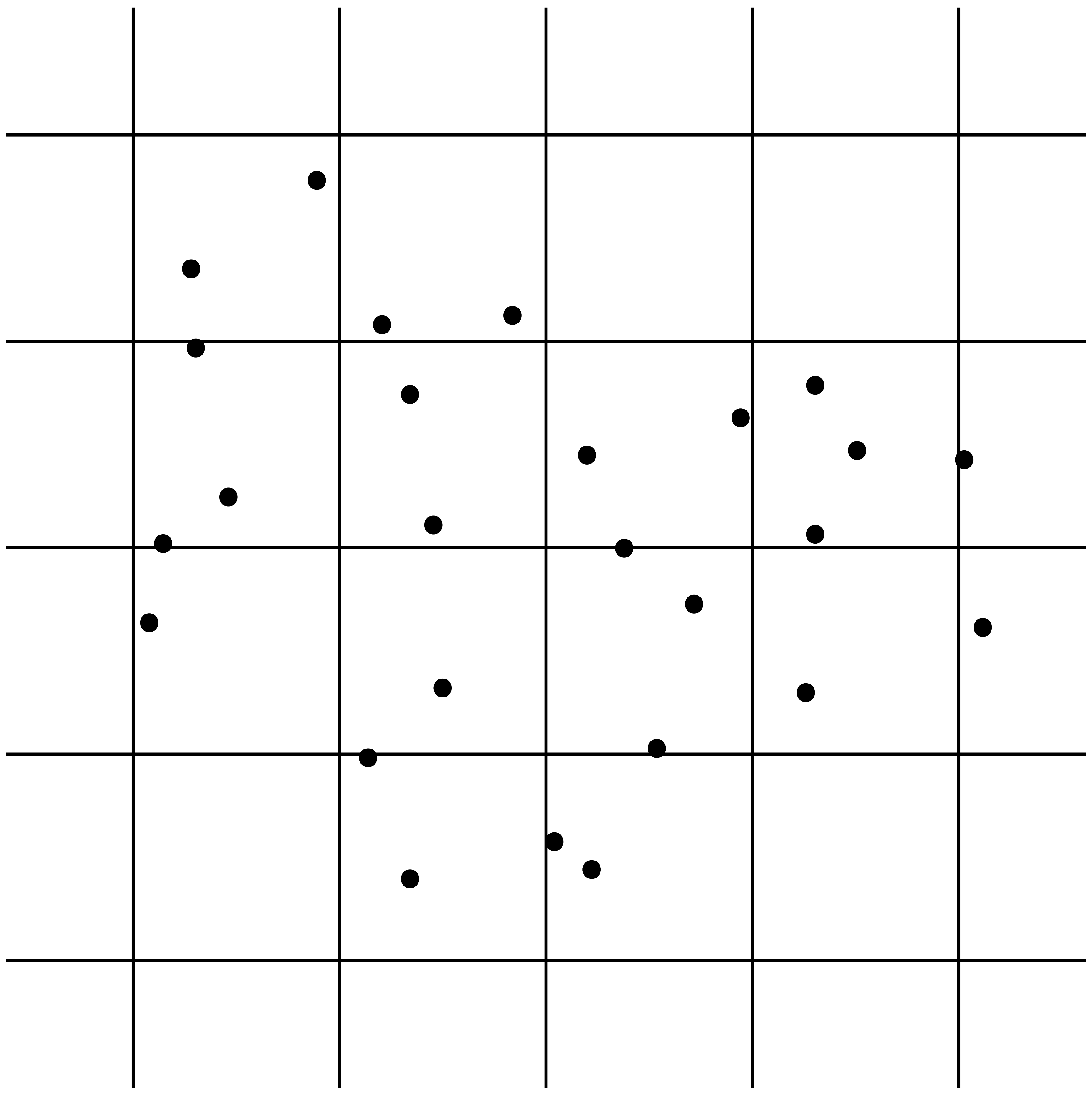


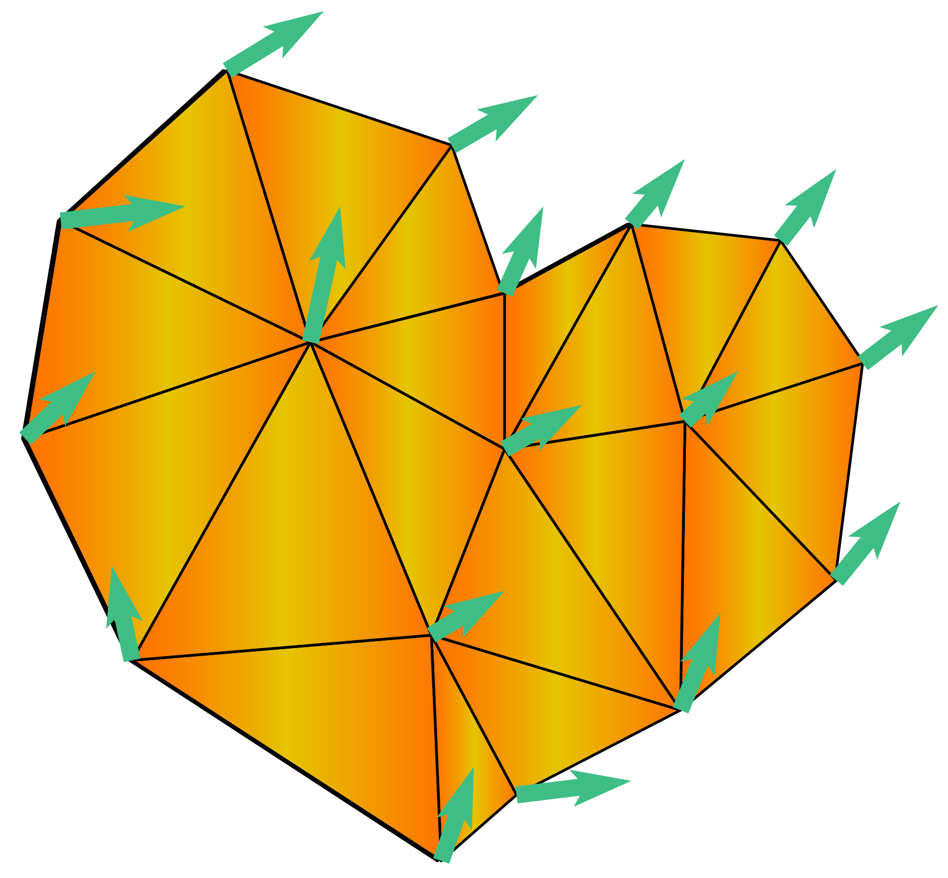
PARTICLE MPM



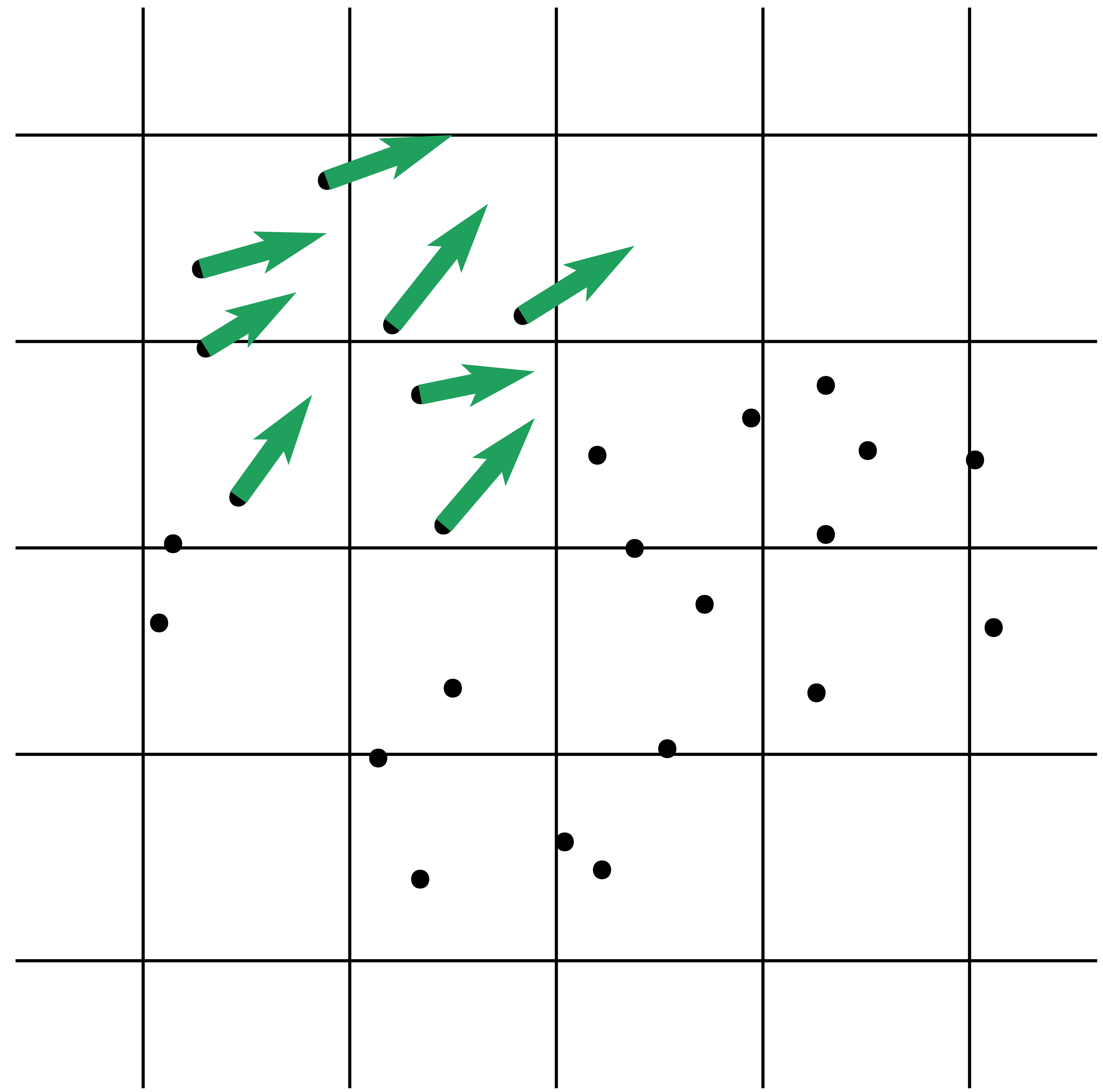


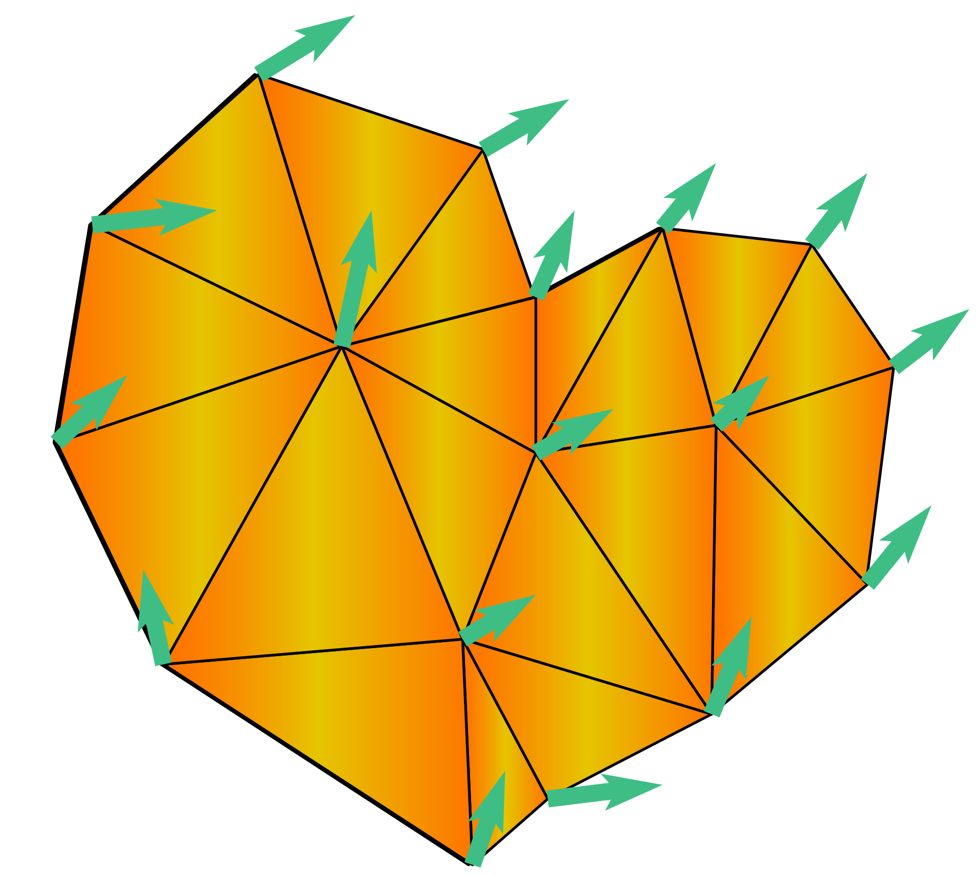
PARTICLE MPM



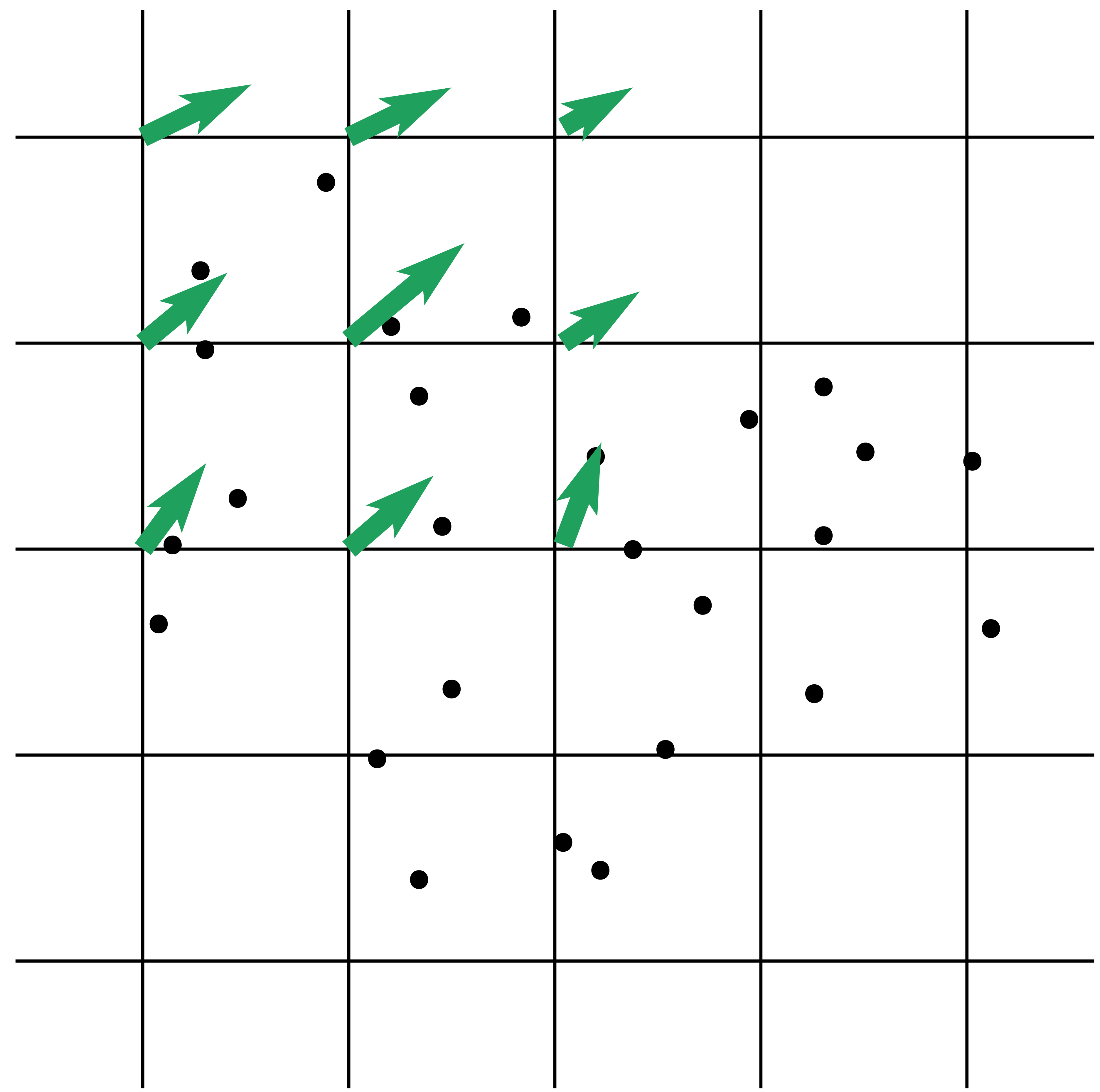


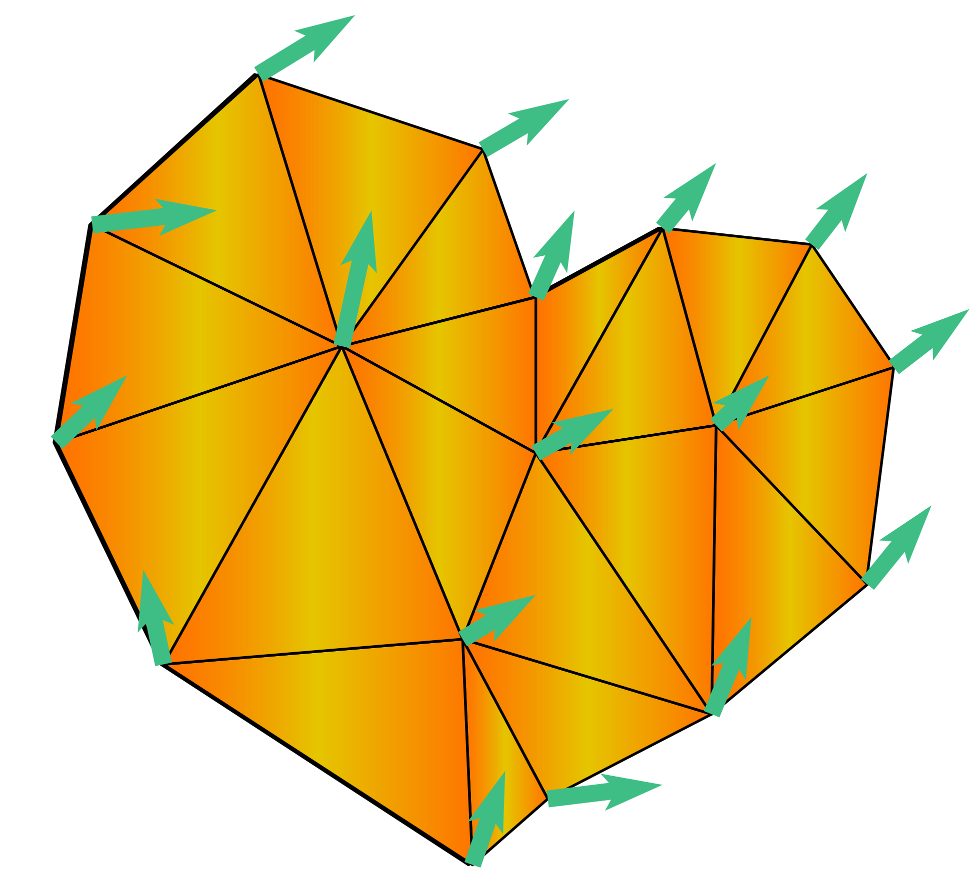
PARTICLE MPM



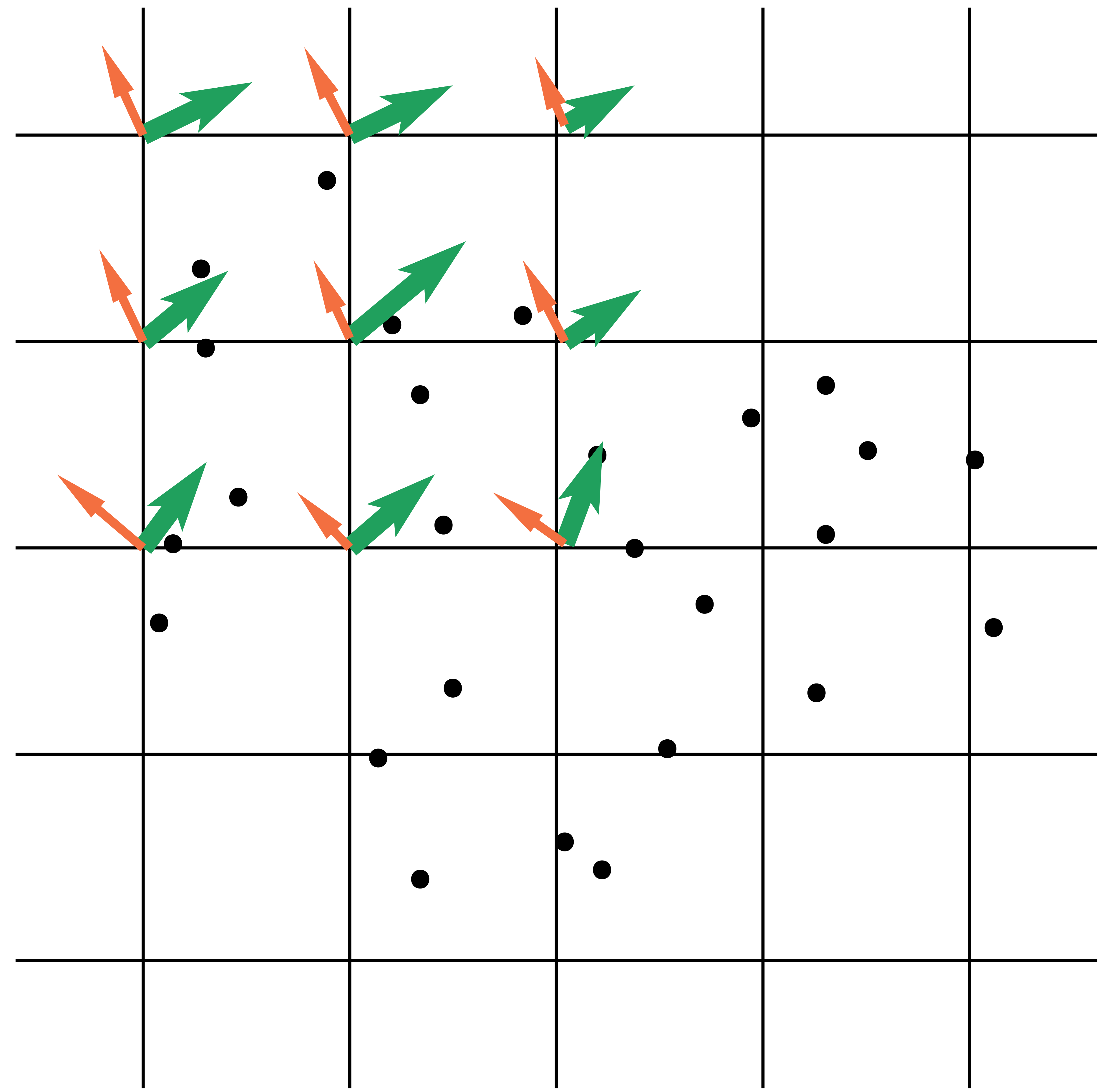


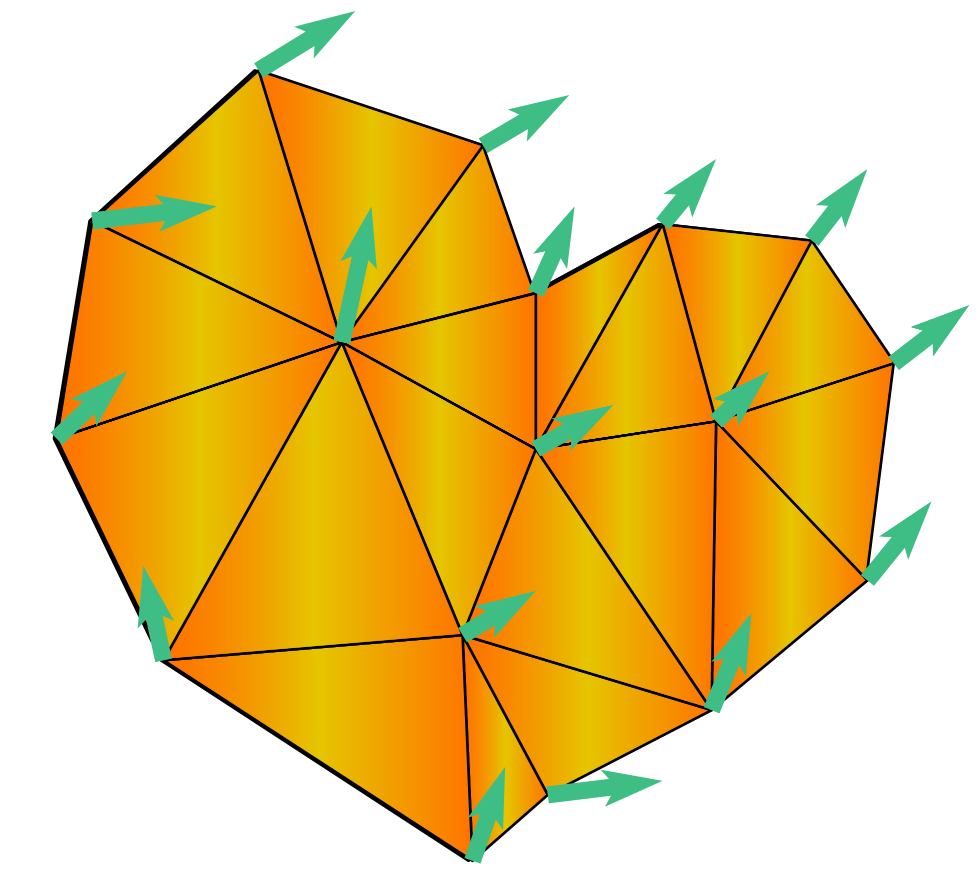
PARTICLE MPM



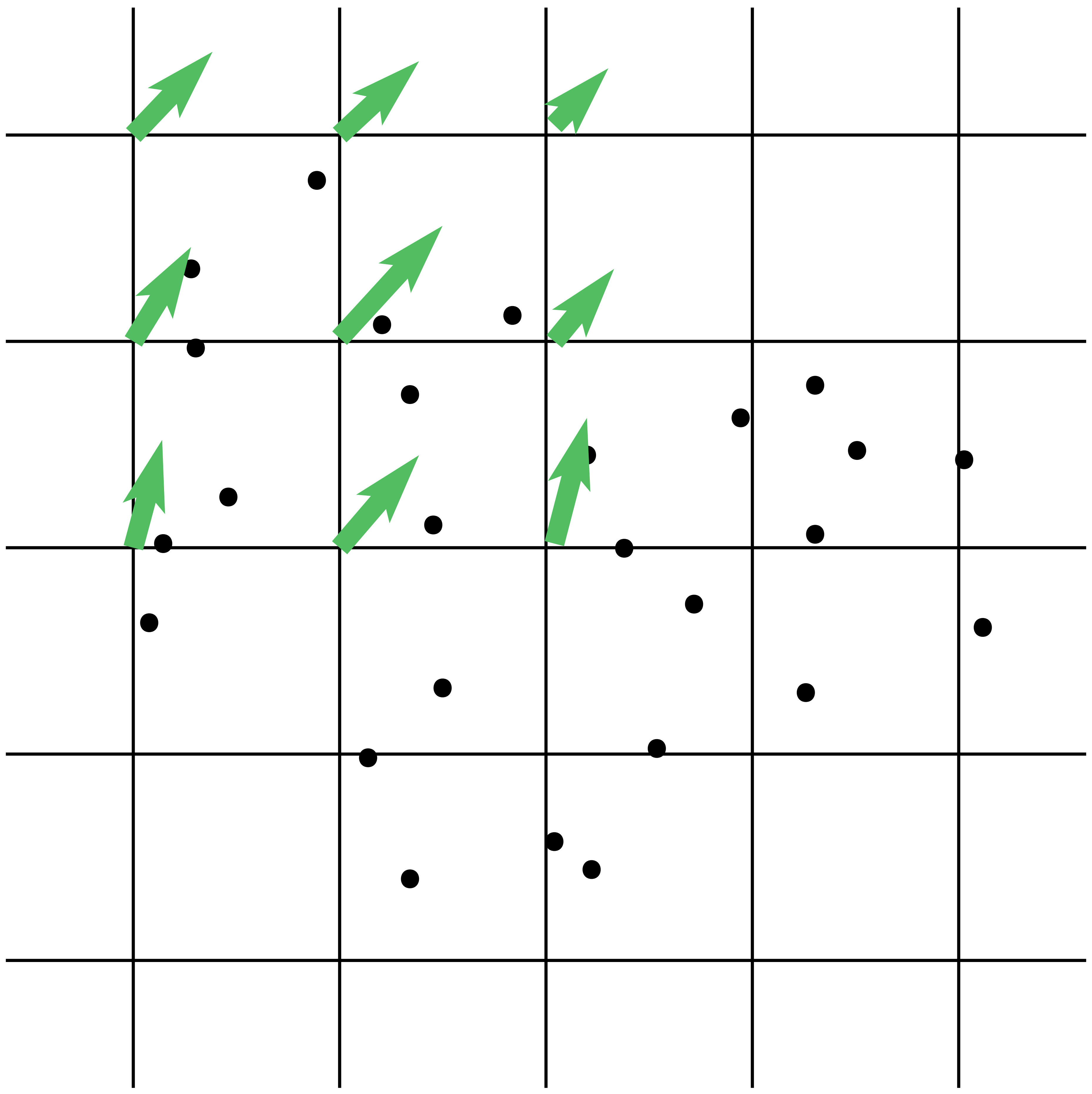


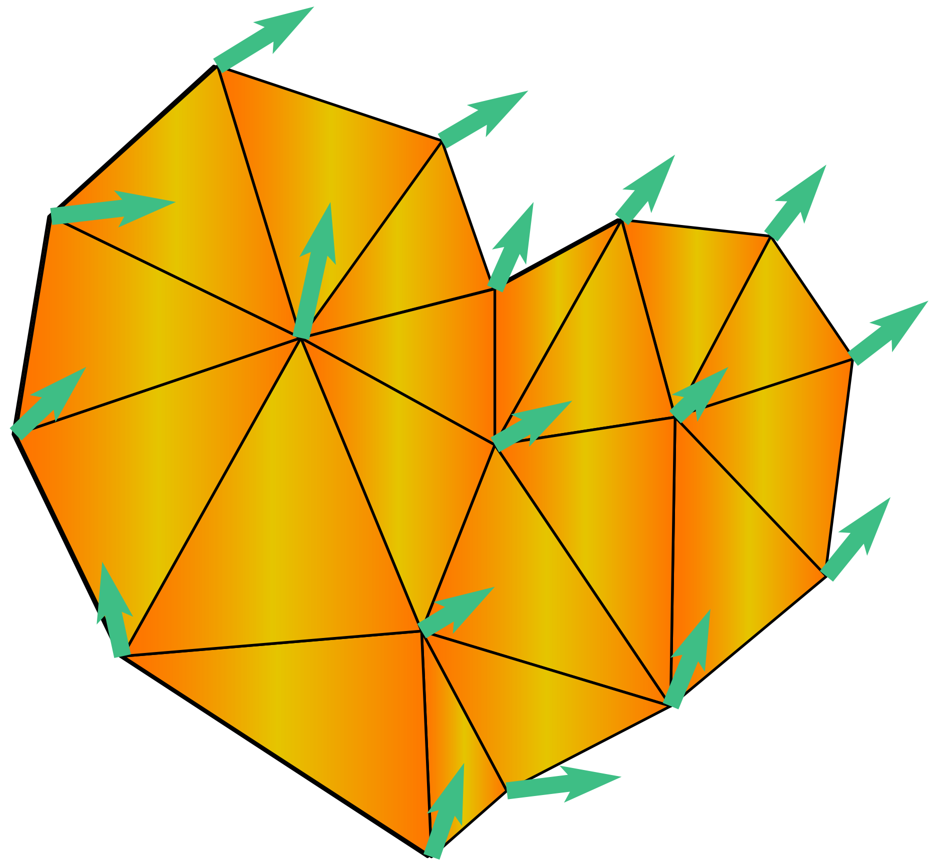
PARTICLE MPM



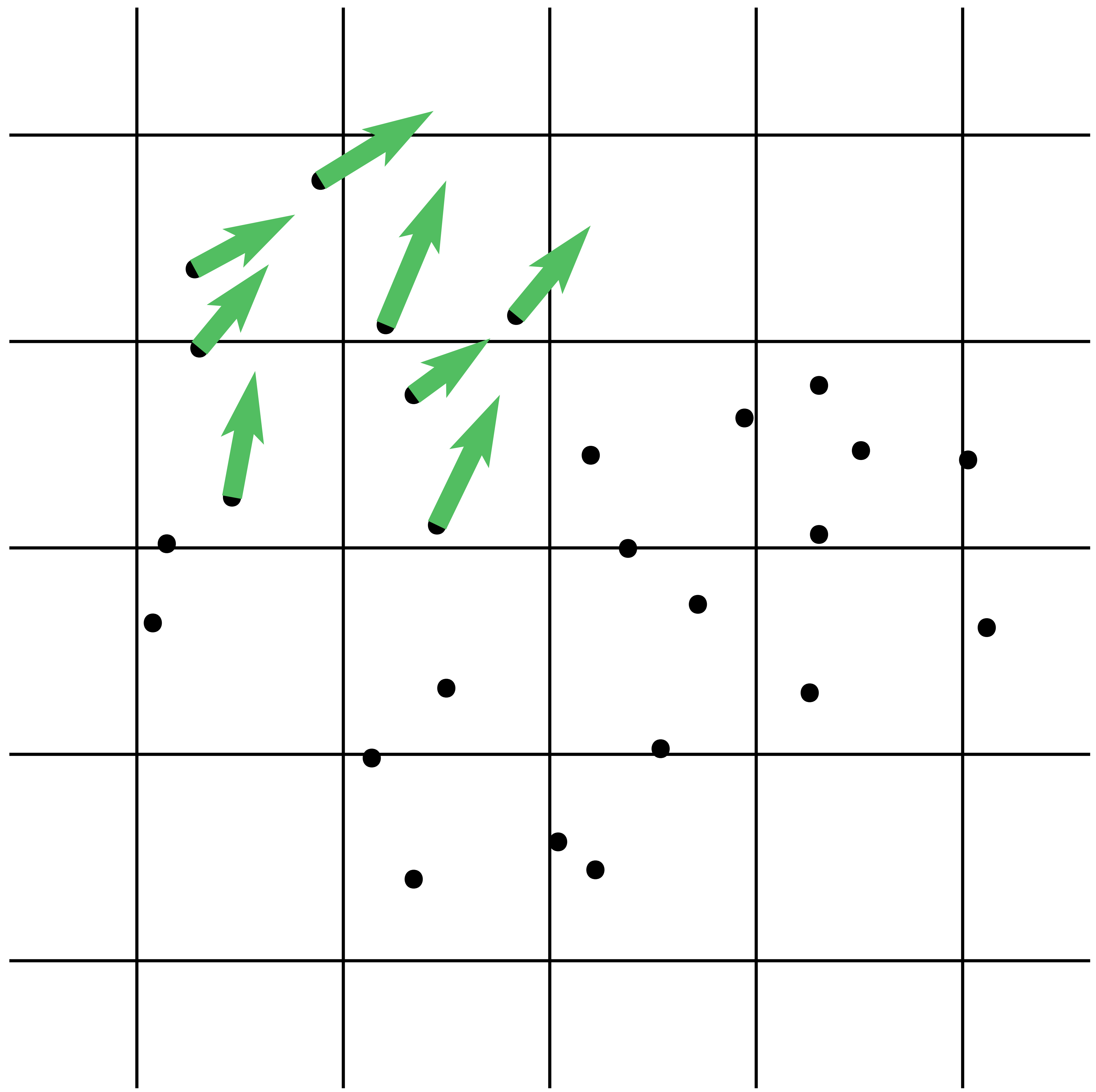


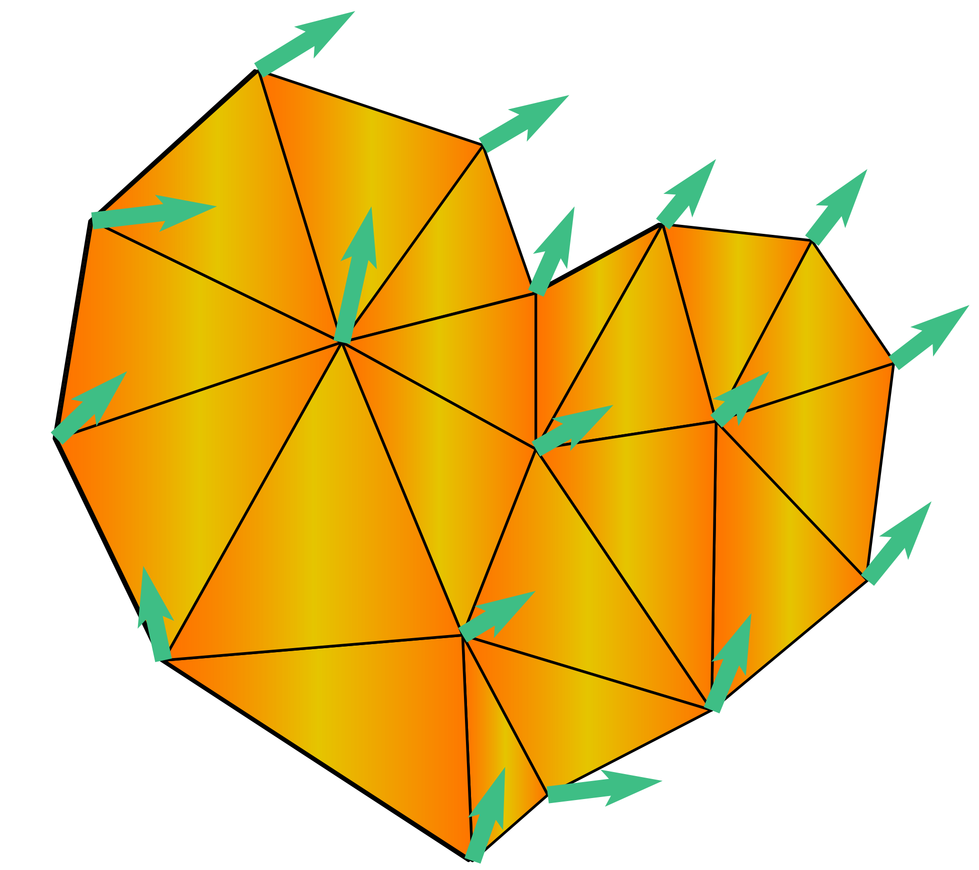
PARTICLE MPM





PARTICLE MPM

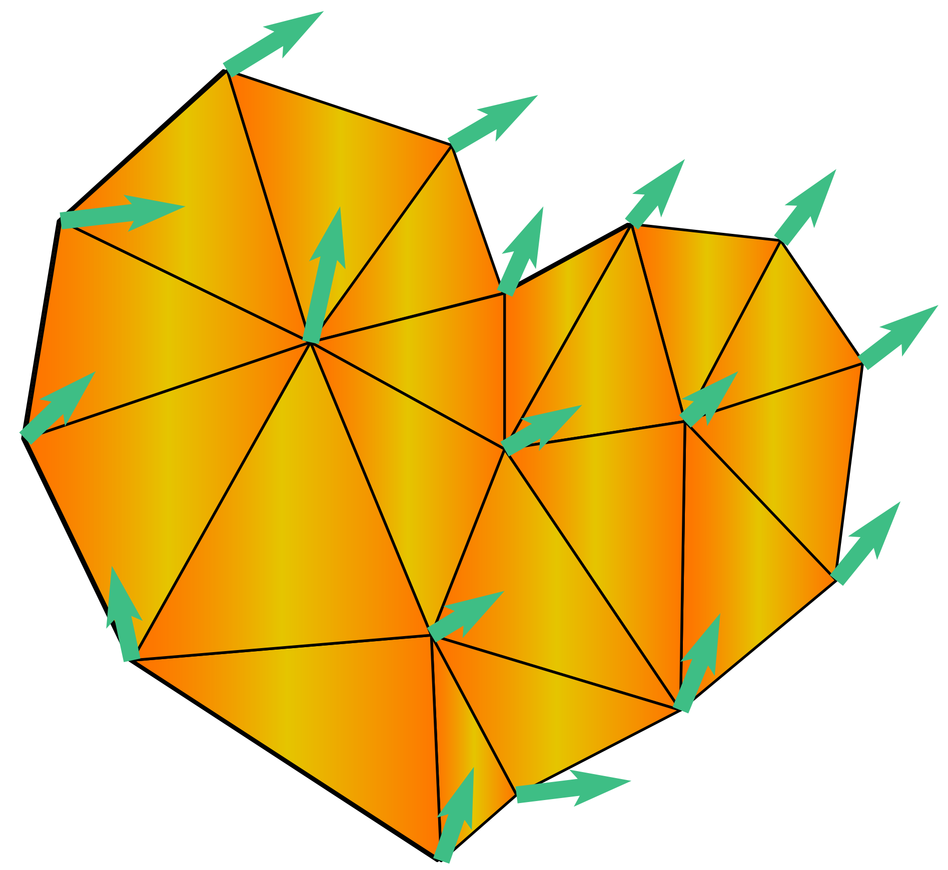




$$\Phi = \sum_p V_p^0 \Psi(\mathbf{F}_p)$$

$$\mathbf{F}_p^{n+1} = \left(\mathbf{I} + \Delta t \sum_i \mathbf{v}_i (\nabla \omega_{ip}^n)^T \right) \mathbf{F}_p^n$$

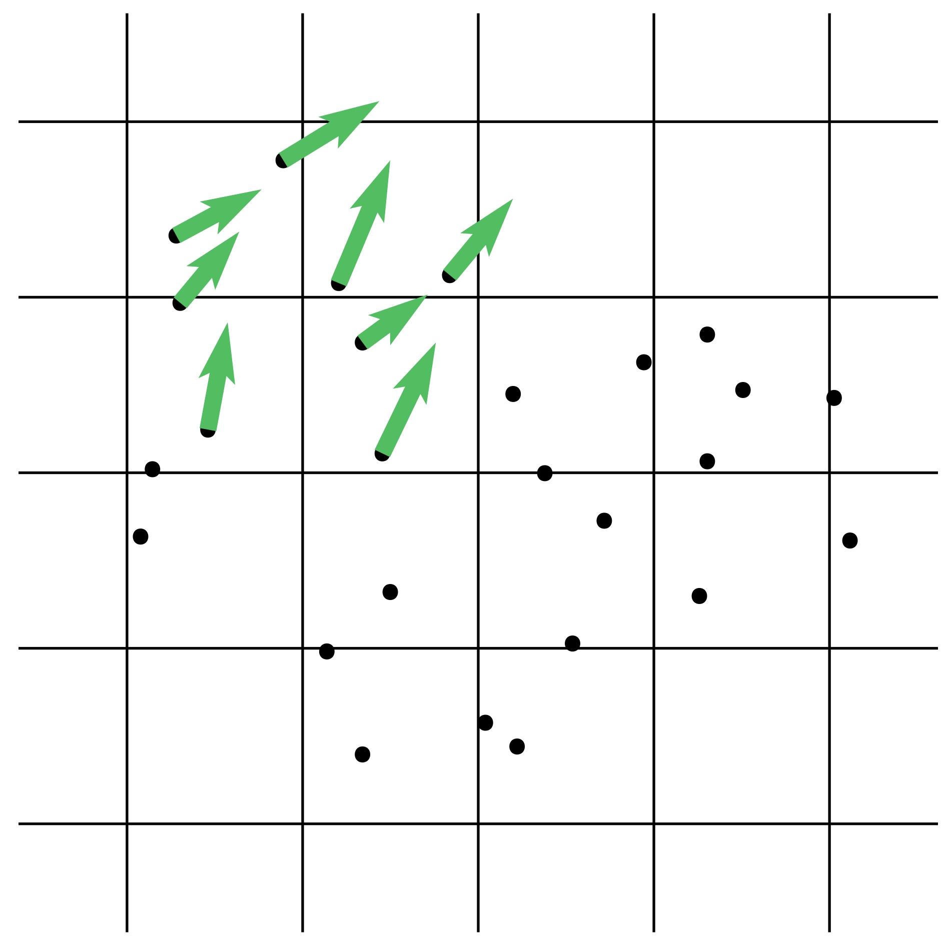
LAGRANGIAN MPM

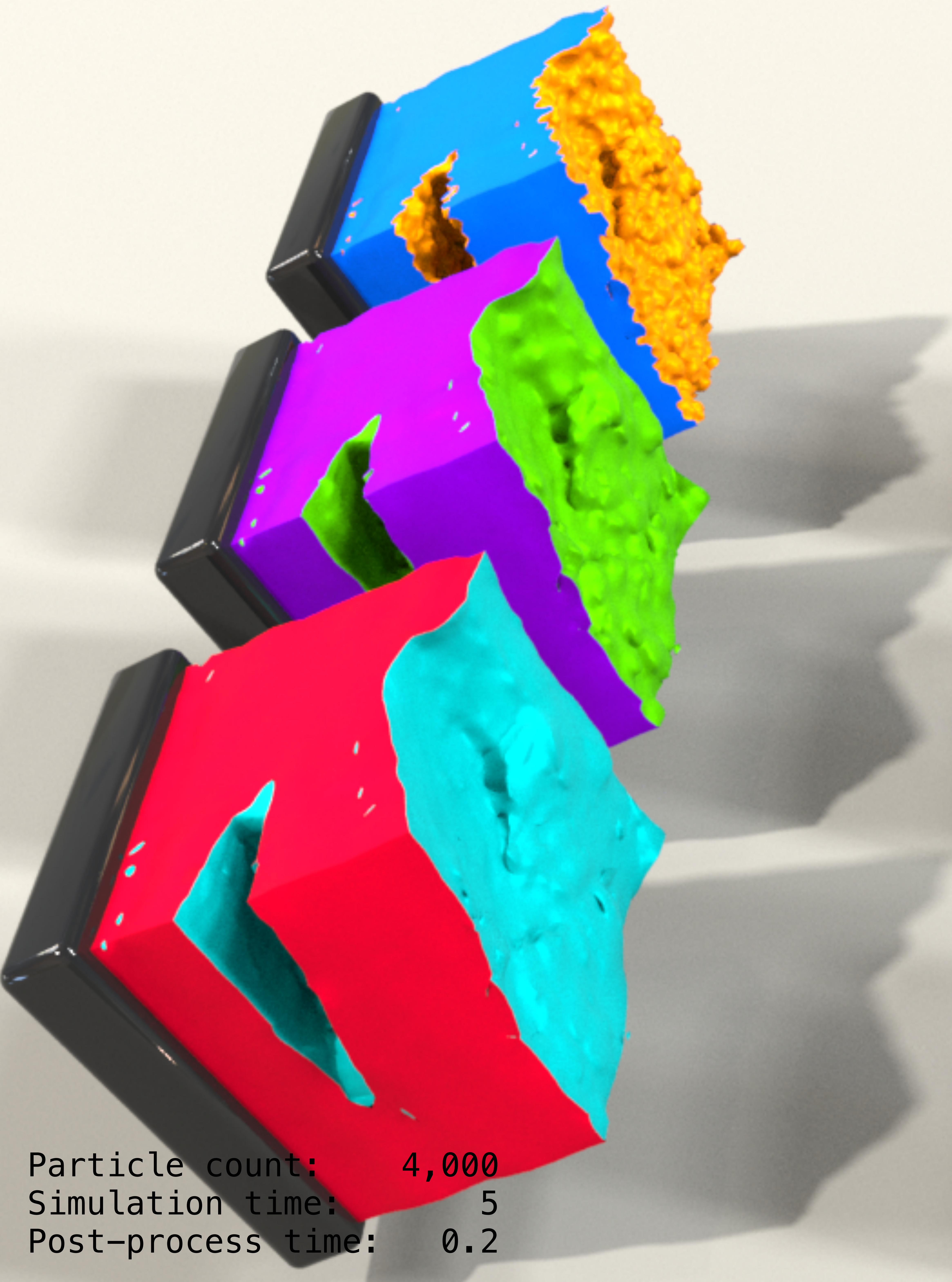


$$\Phi = \sum_e V_e^0 \Psi(\mathbf{F}_e)$$

$$\mathbf{F}_e^n = \sum_q \mathbf{x}_q^n \nabla N_q(\mathbf{X}_e)^T$$

$$\mathbf{f}_i^n = \sum_q \omega_{iq}^n \mathbf{f}_q^n$$





SIMULATION AND VISUALIZATION OF DUCTILE FRACTURE

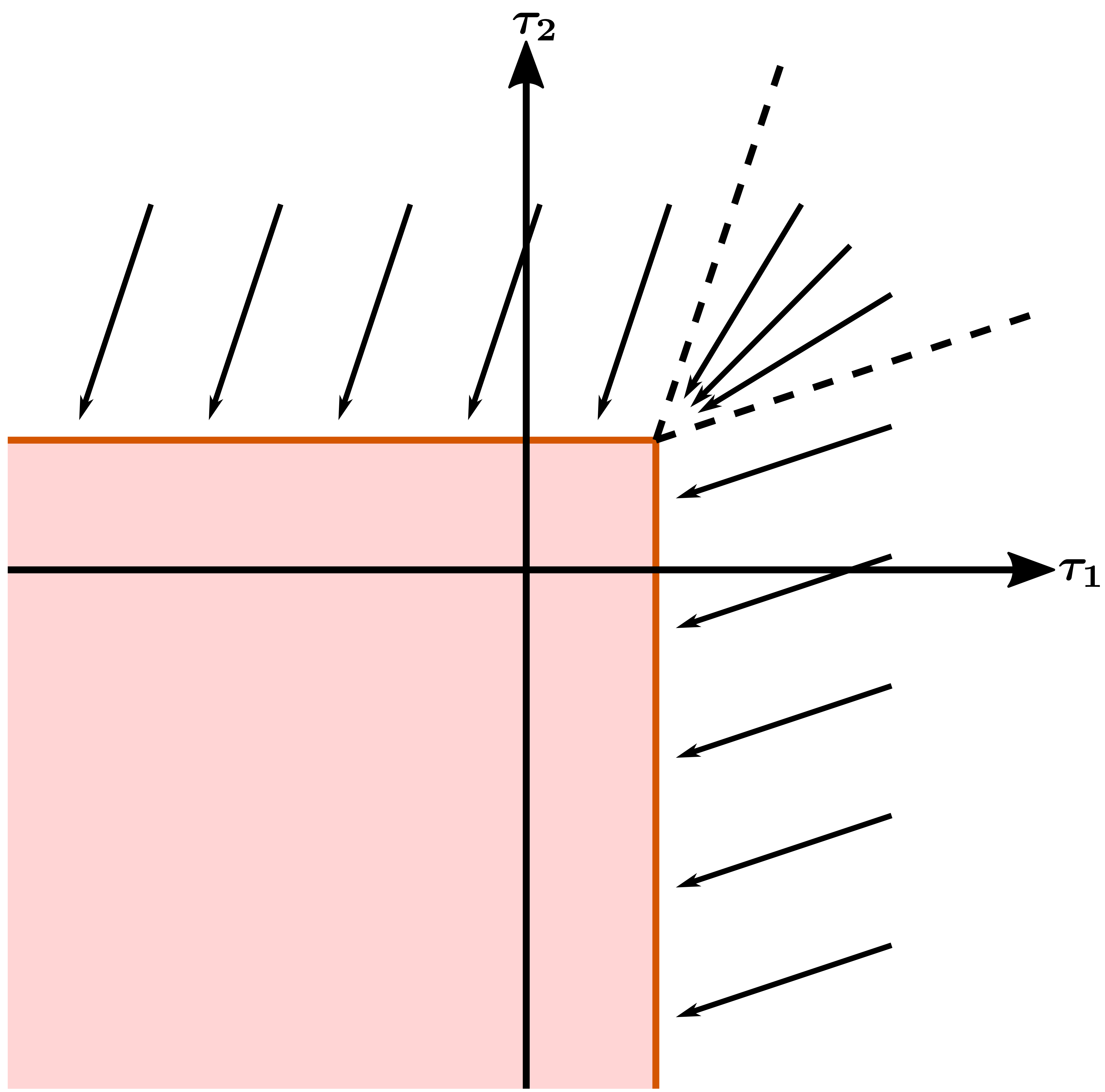
RANKINE YIELD SURFACE [MÜLLER ET AL. 2014]

- Constraining maximal principal stress

$$y(\tau) = \max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} \mathbf{u}^T \tau \mathbf{v} - \tau_C \leq 0$$

- Mode I yielding (tension)
- Softening rule

$$\tau_C^{n+1} = \tau_C^n + \alpha \left(\max_{\|\mathbf{u}\|=\|\mathbf{v}\|=1} \mathbf{u}^T \epsilon^{n+1} \mathbf{v} - \max_{\|\tilde{\mathbf{u}}\|=\|\tilde{\mathbf{v}}\|=1} \tilde{\mathbf{u}}^T \epsilon^{tr} \tilde{\mathbf{v}} \right)$$



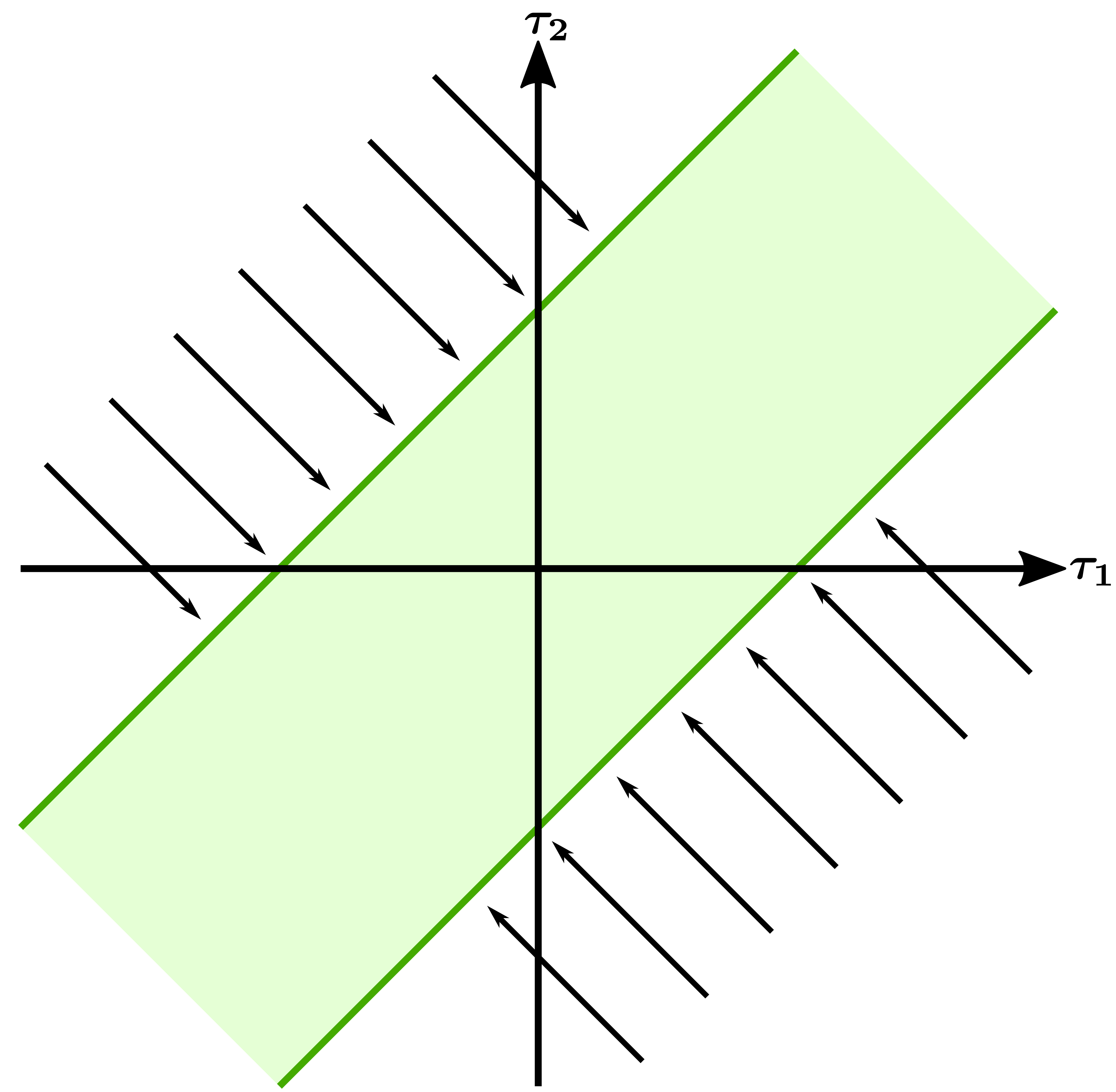

Particle count: 130,000
Simulation time: 15
Post-process time: 8

VON MISES (J2) YIELD SURFACE

- Constraining shear stress

$$y(\tau) = \|\tau - \mathrm{tr}(\tau)\mathbf{I}\|_F - \tau_C \leq 0$$

- Mode II and III yielding (shear)
- Softening can be added



$\tau_C/E = 1$
$\tau_C/E = 0.7$
$\tau_C/E = 0.5$
Particle count: 60,000
Simulation time: 11
Post-process time: 4

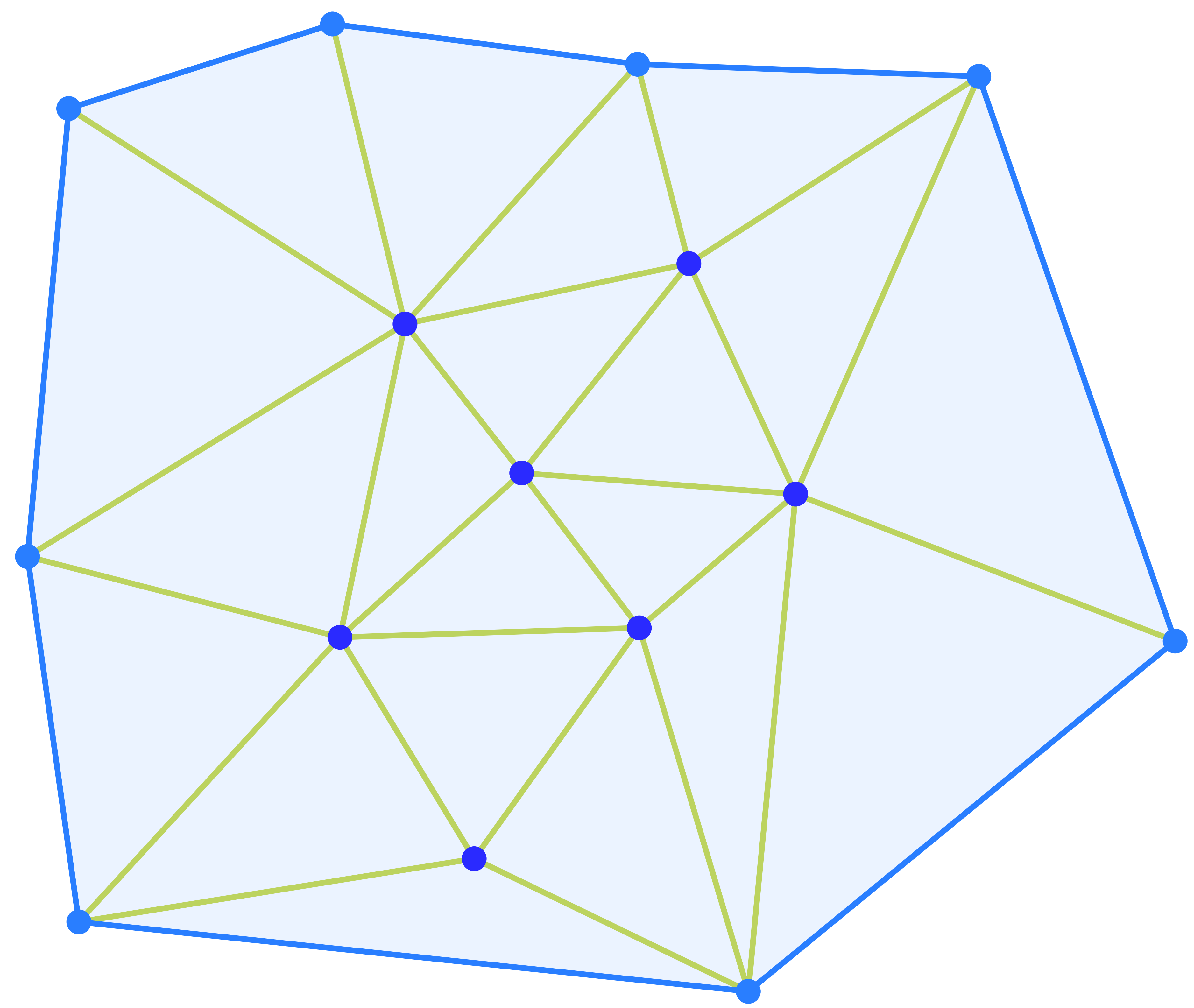
Particle count: 60,000
Simulation time: 11
Post-process time: 5

THREE STEPS OF CREATING FRACTURING MESH

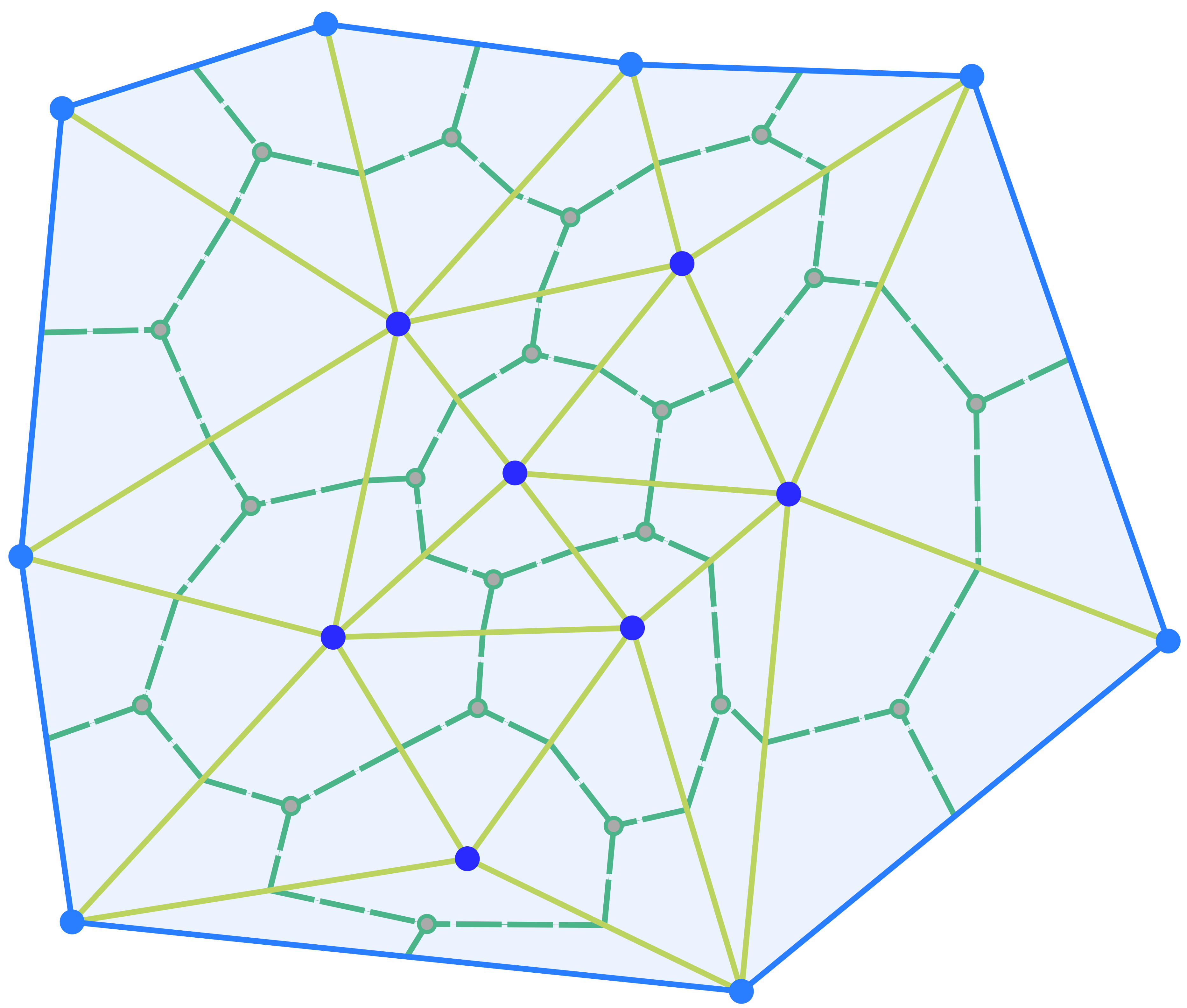
- Fracturing topology (that evolves with time)
- Extrapolate positions for the added vertices
- Smoothing crack surface to reduce mesh-dependent noise
- Advantage: per-frame post-process instead of per-time-step treatment

FRACTURING TOPOLOGY

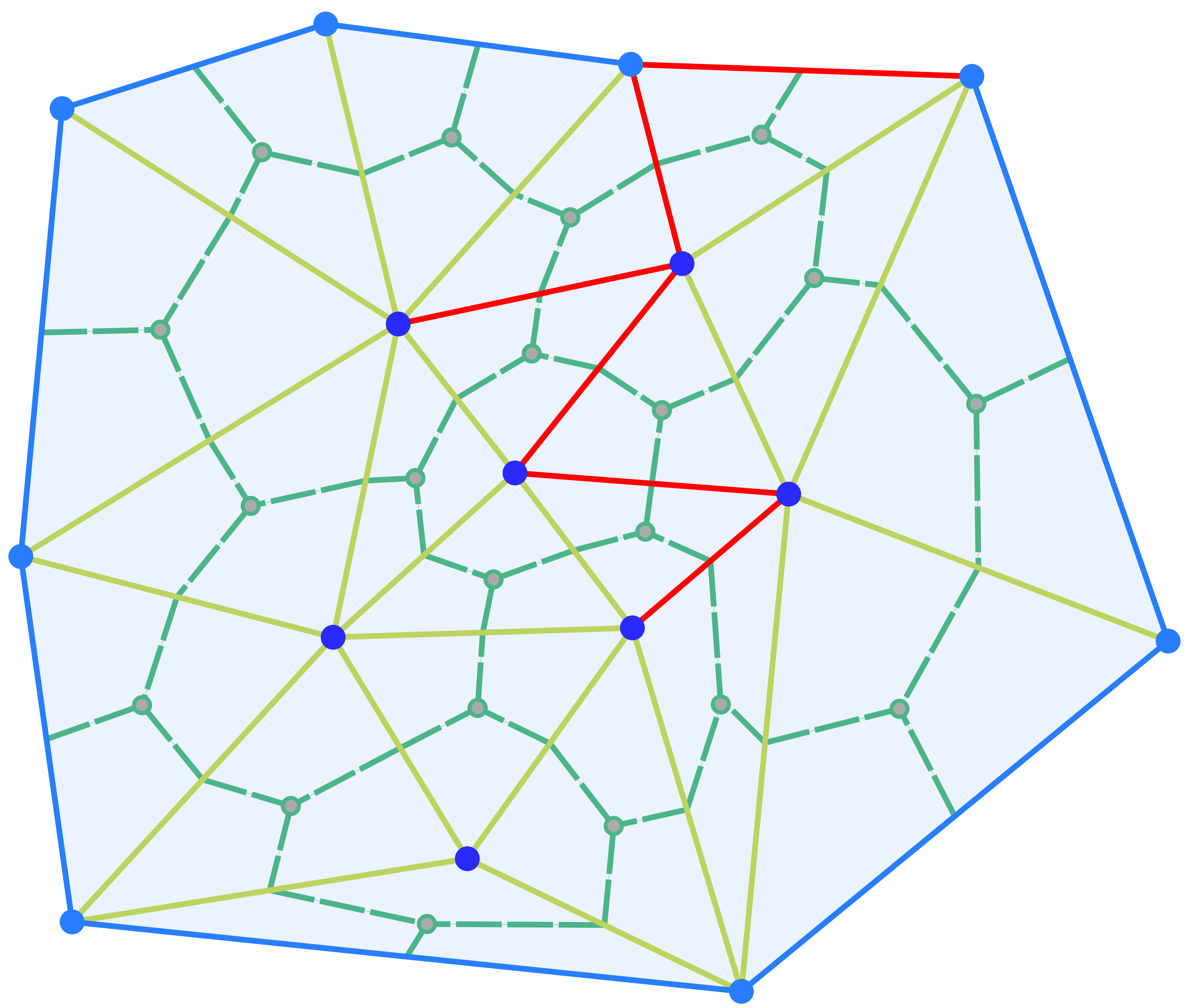
FRACTURING TOPOLOGY



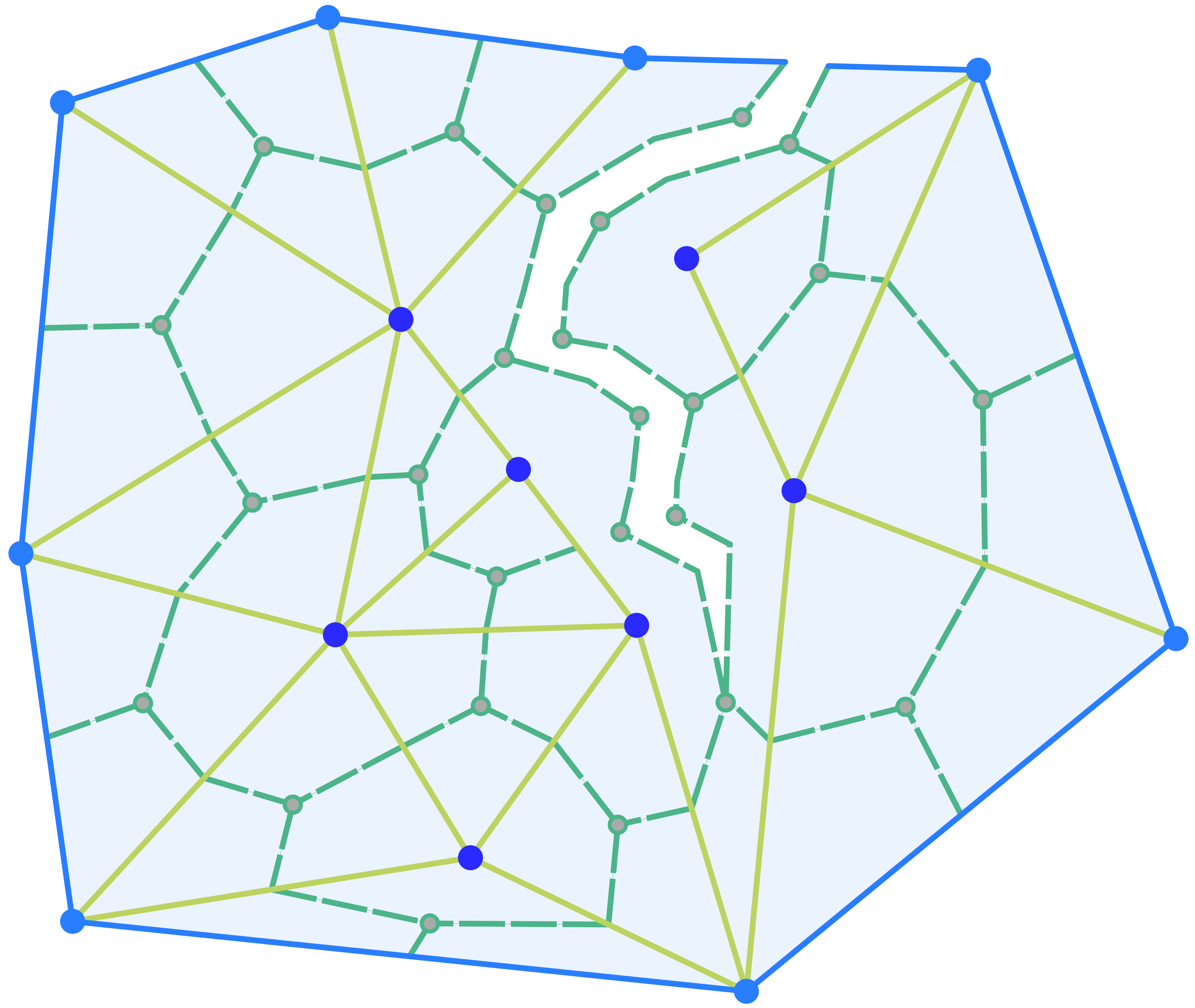
FRACTURING TOPOLOGY



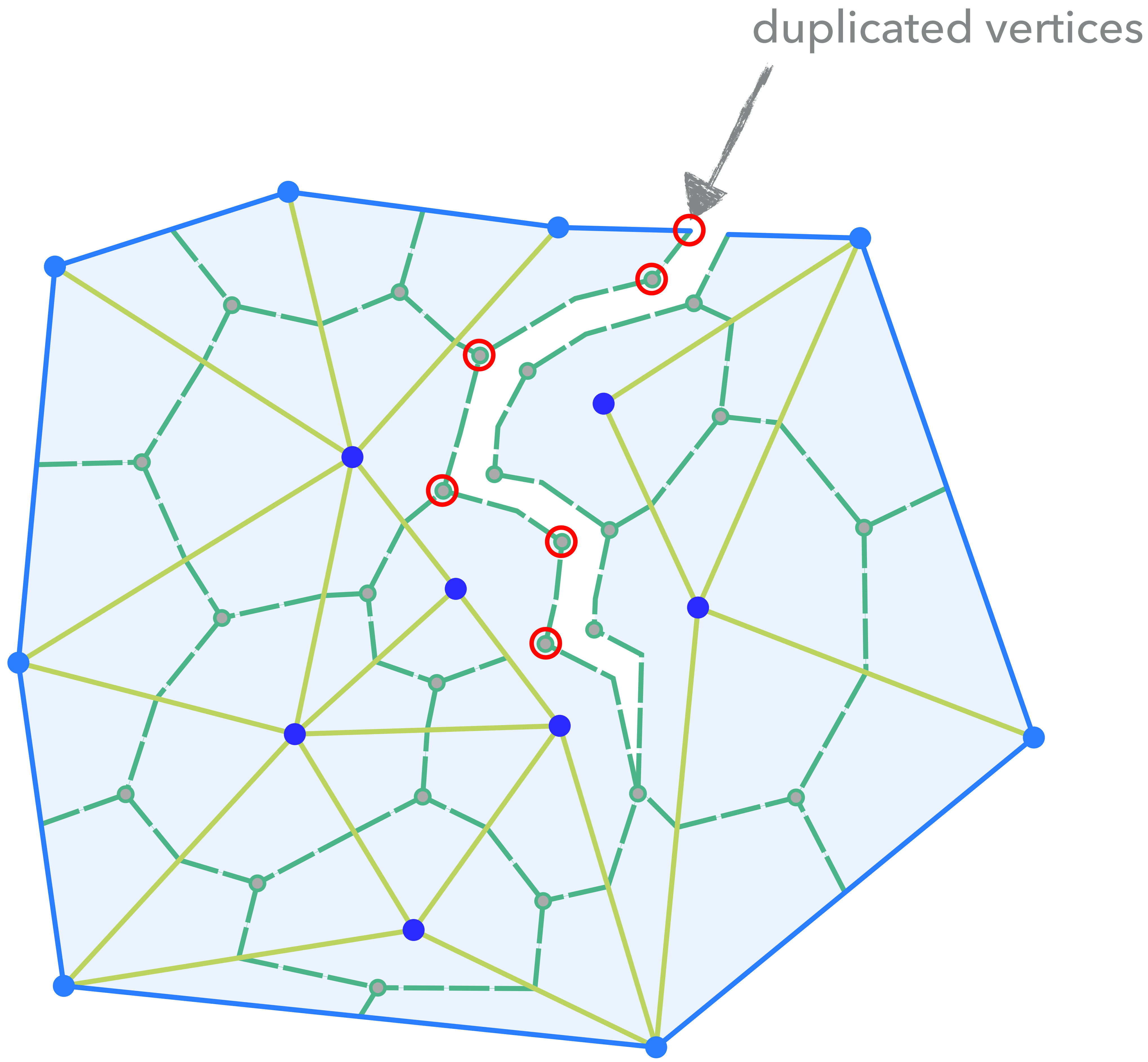
FRACTURING TOPOLOGY



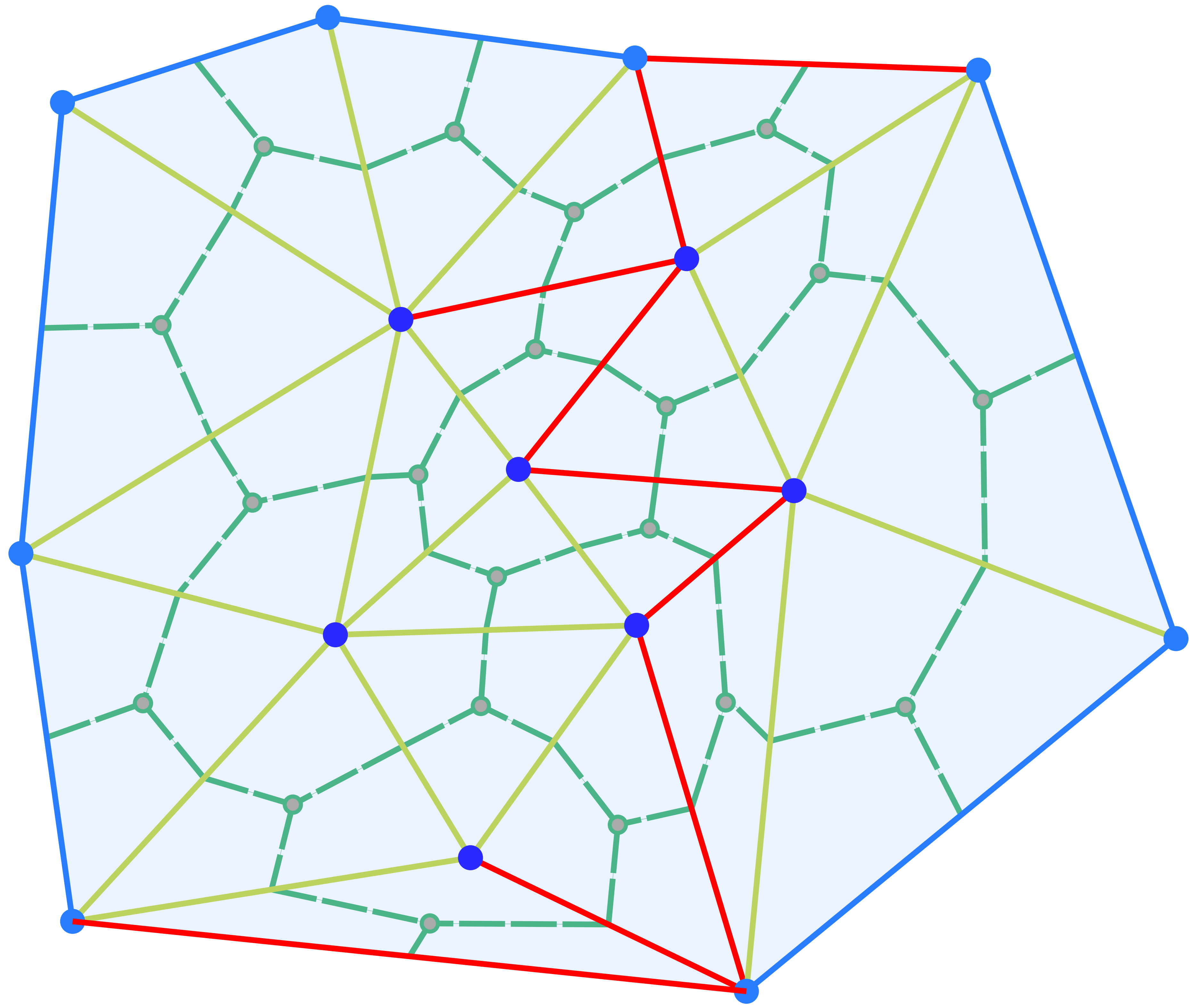
FRACTURING TOPOLOGY



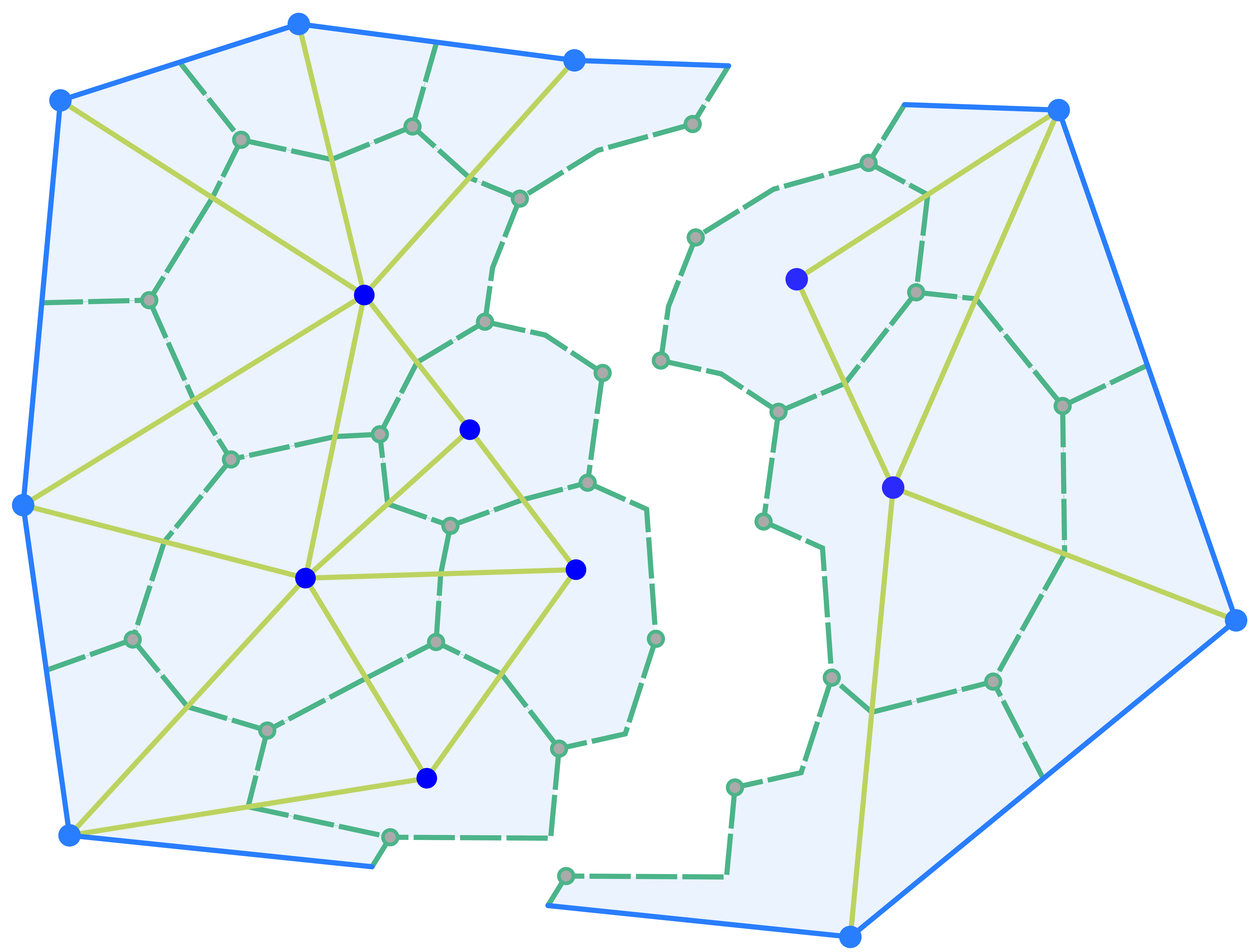
FRACTURING TOPOLOGY



FRACTURING TOPOLOGY

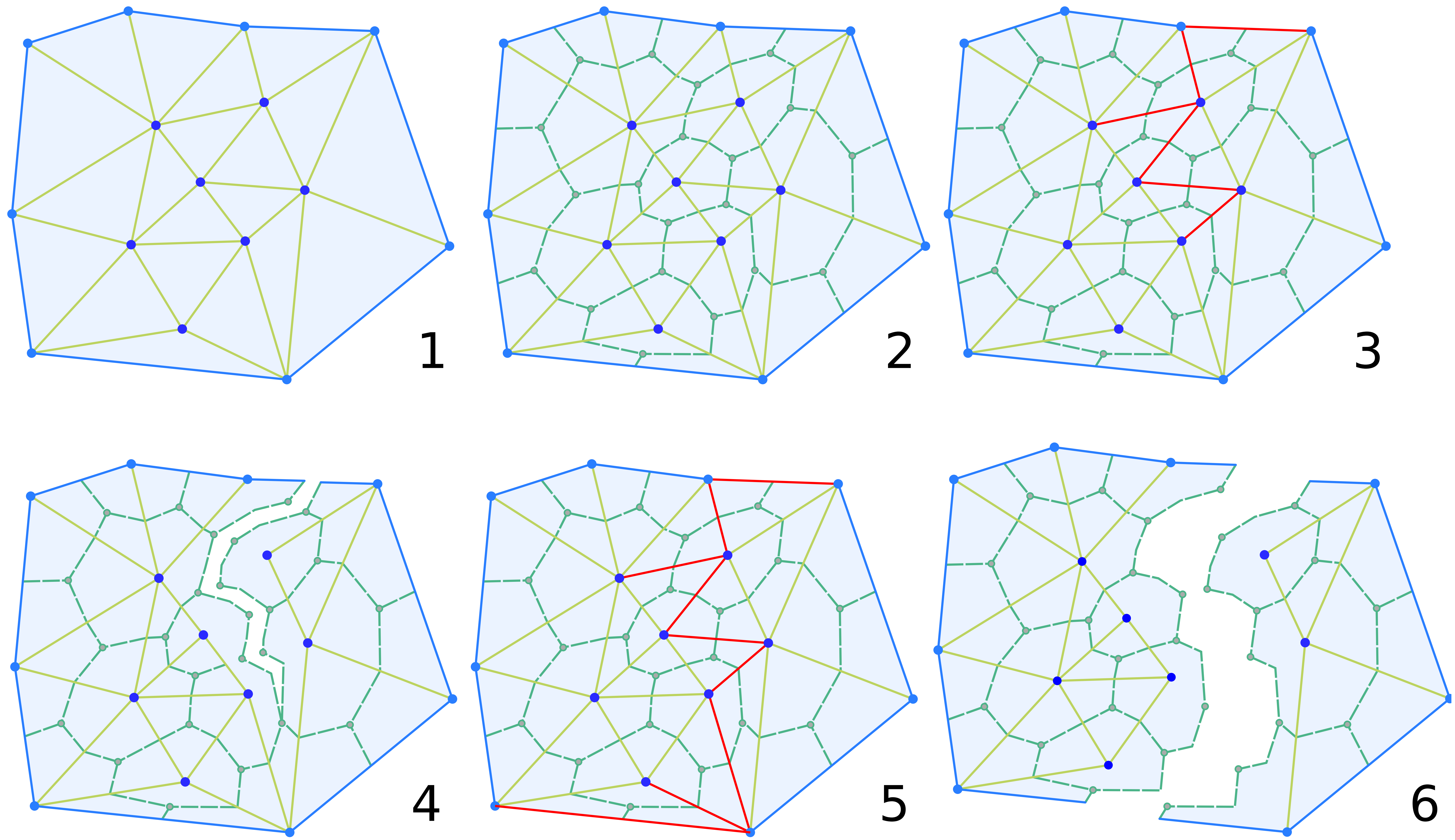


FRACTURING TOPOLOGY



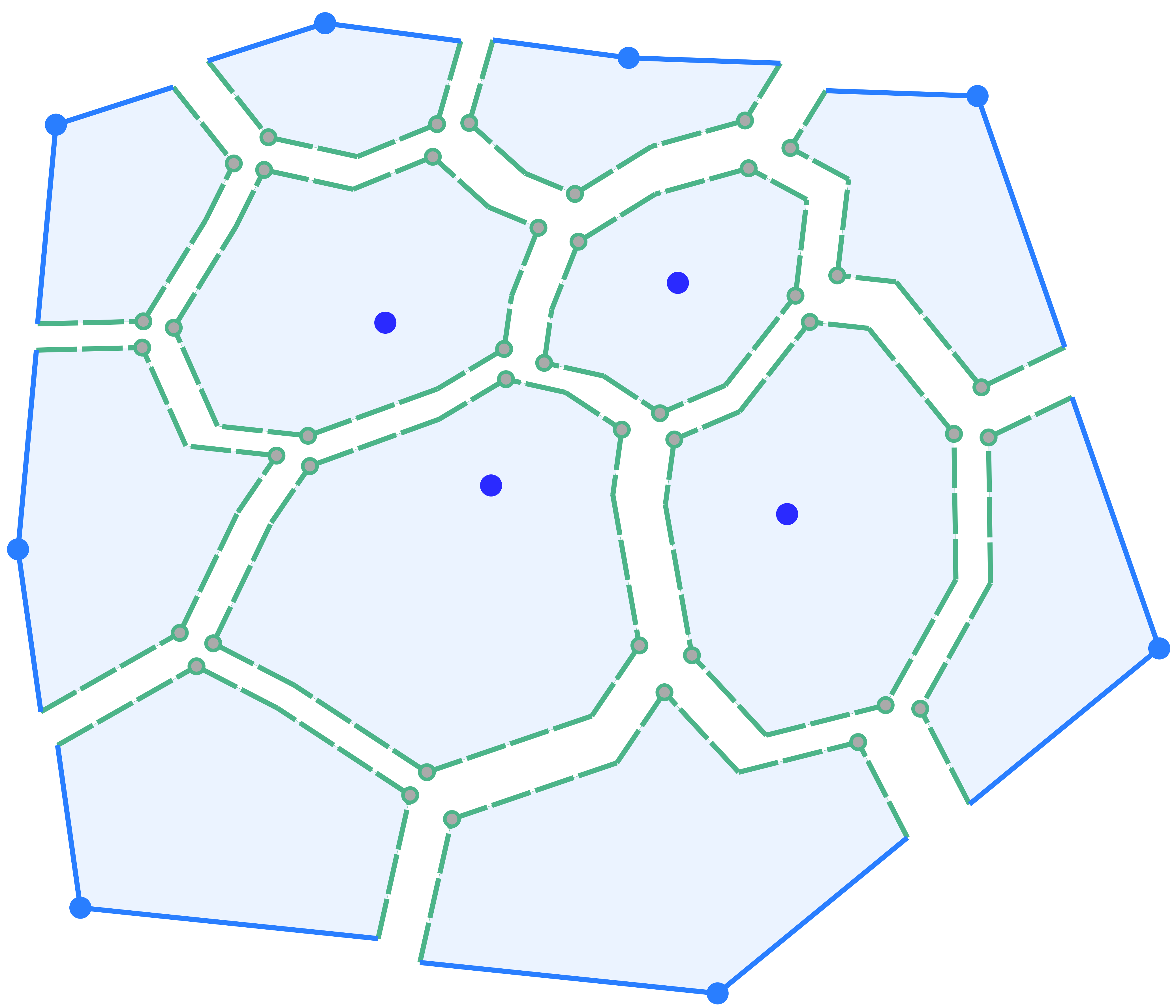
FRACTURING TOPOLOGY

- Subdivided mesh
- Edge-stretching cutting criterion
- Evolves with time

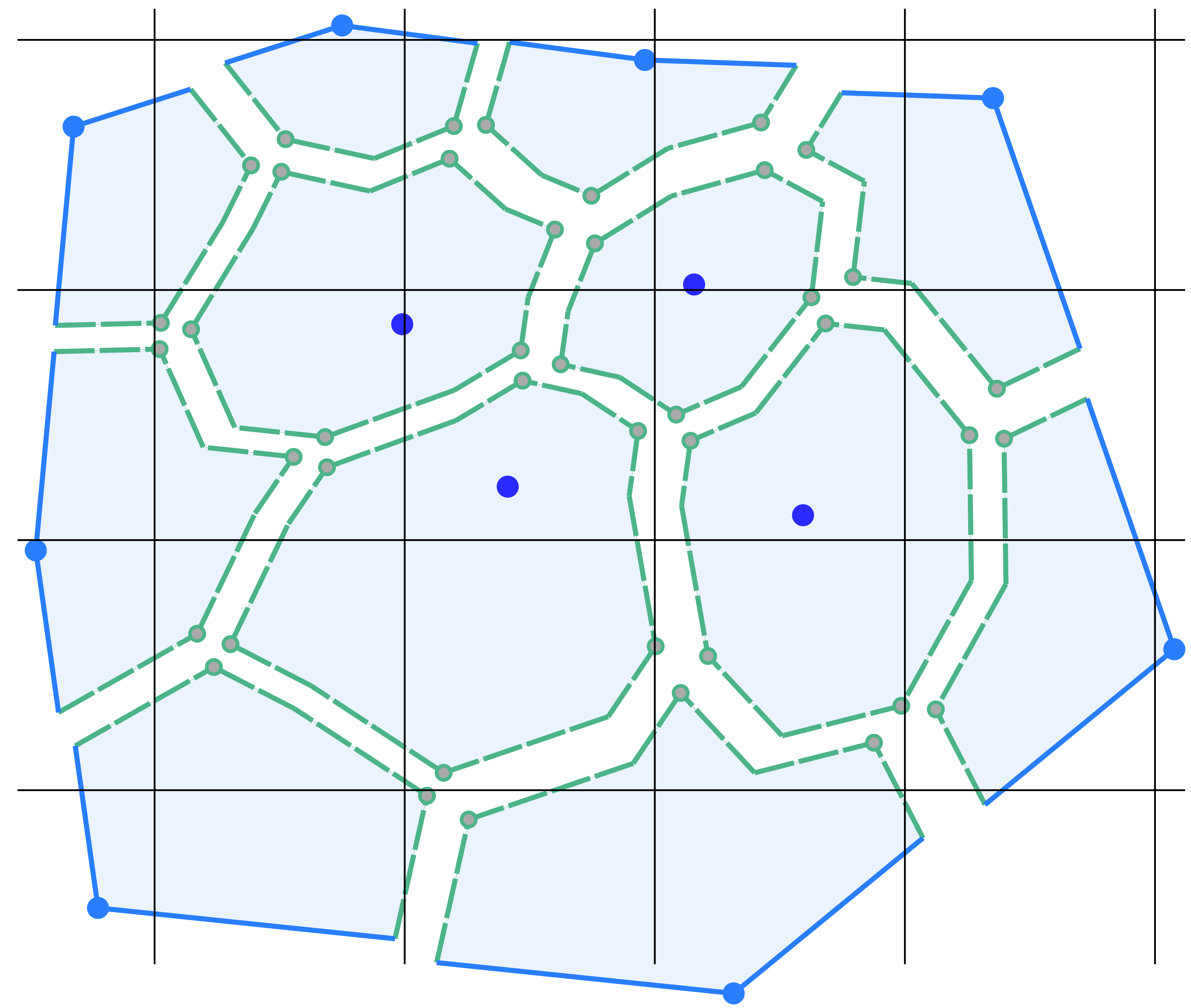


EXTRAPOLATING POSITIONS FOR ADDED VERTICES

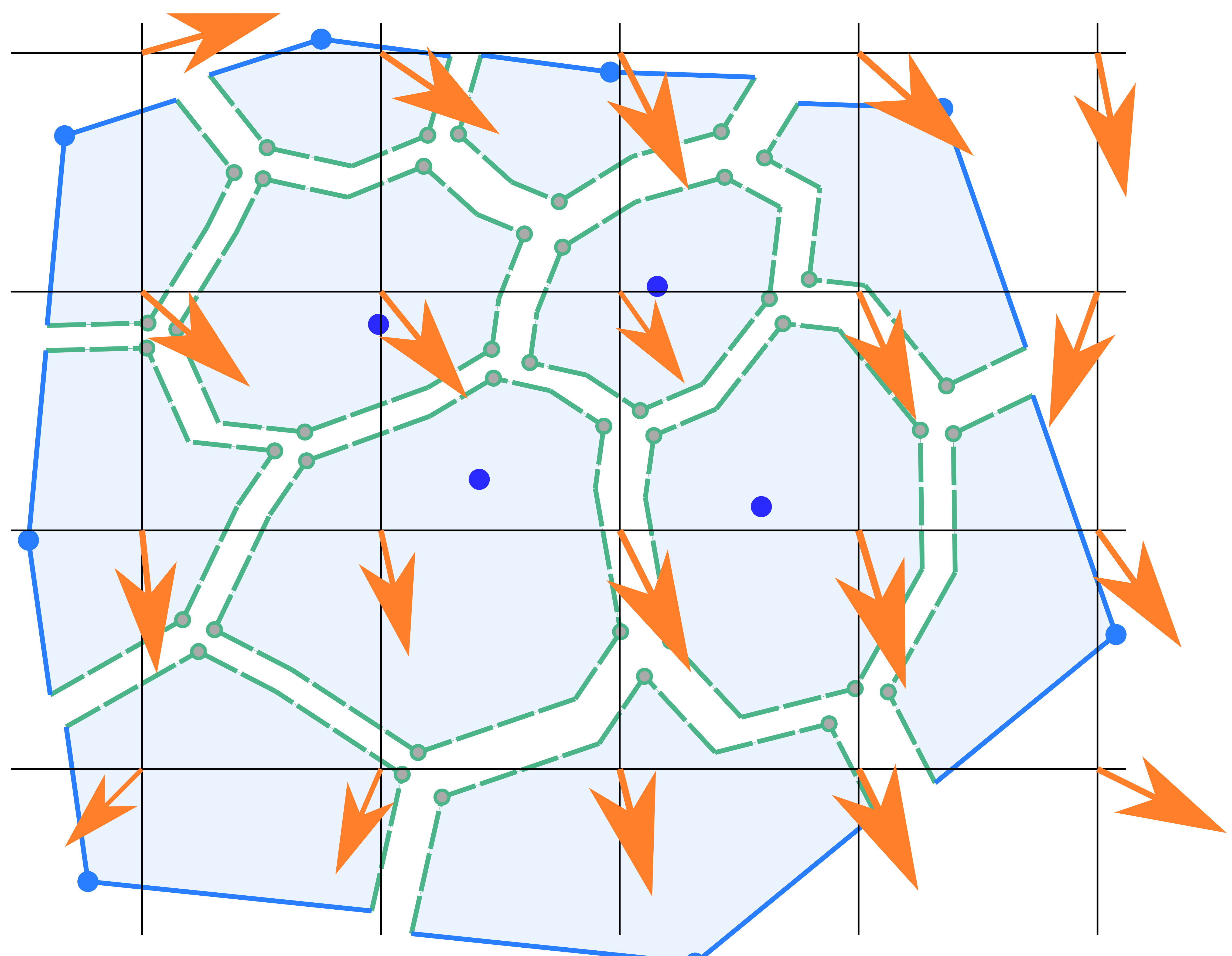
EXTRAPOLATING POSITIONS FOR ADDED VERTICES



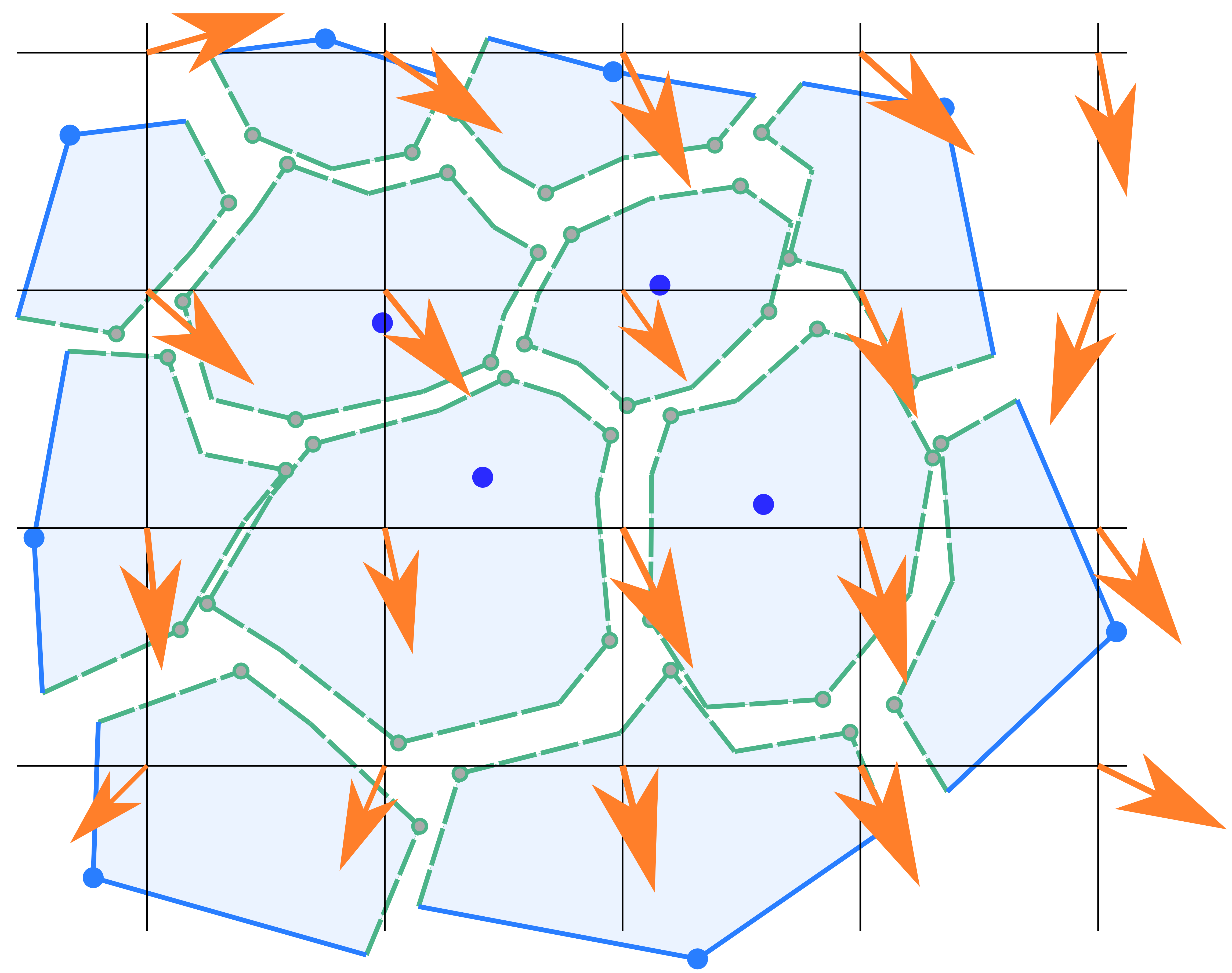
EXTRAPOLATING POSITIONS FOR ADDED VERTICES



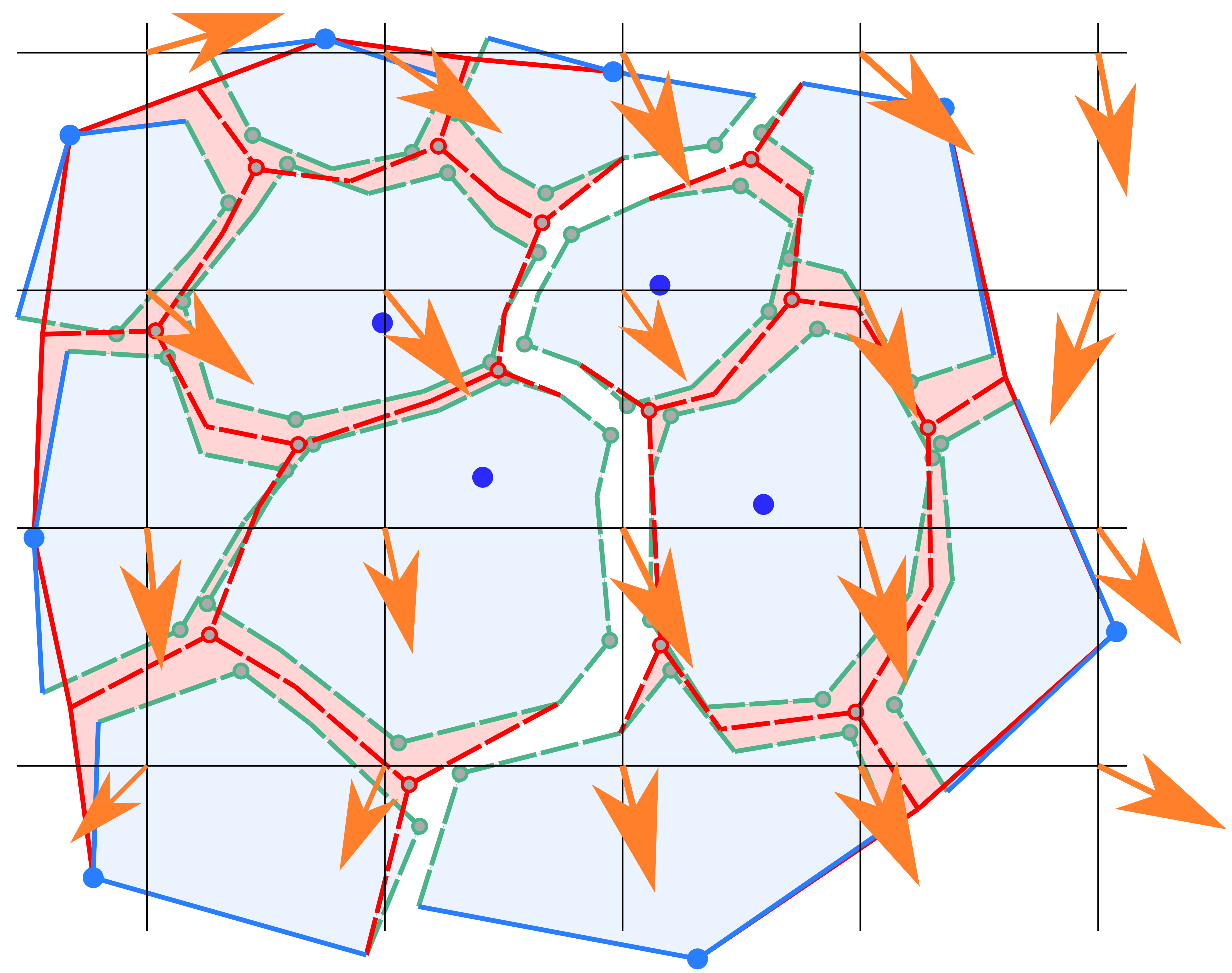
EXTRAPOLATING POSITIONS FOR ADDED VERTICES



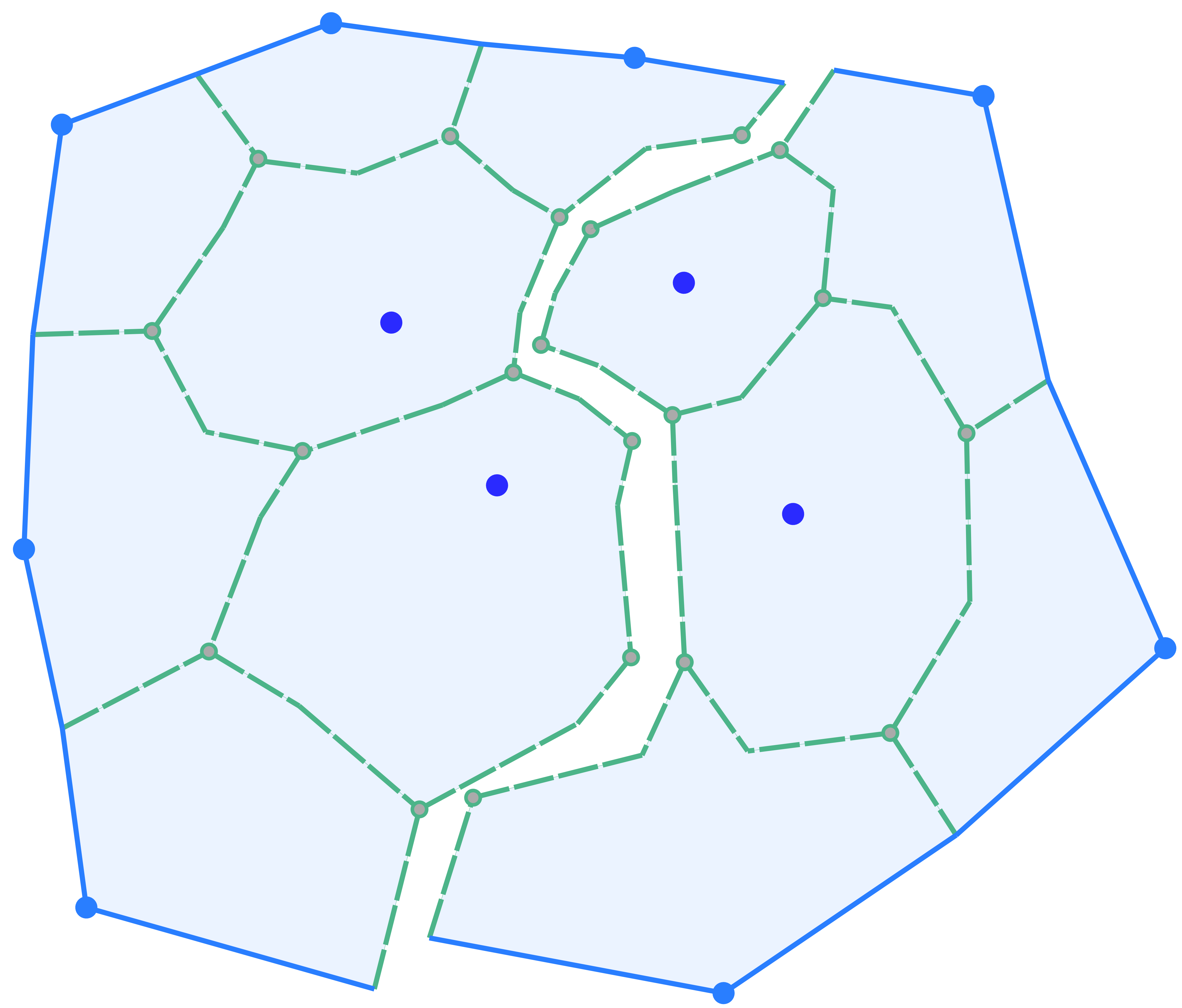
EXTRAPOLATING POSITIONS FOR ADDED VERTICES



EXTRAPOLATING POSITIONS FOR ADDED VERTICES

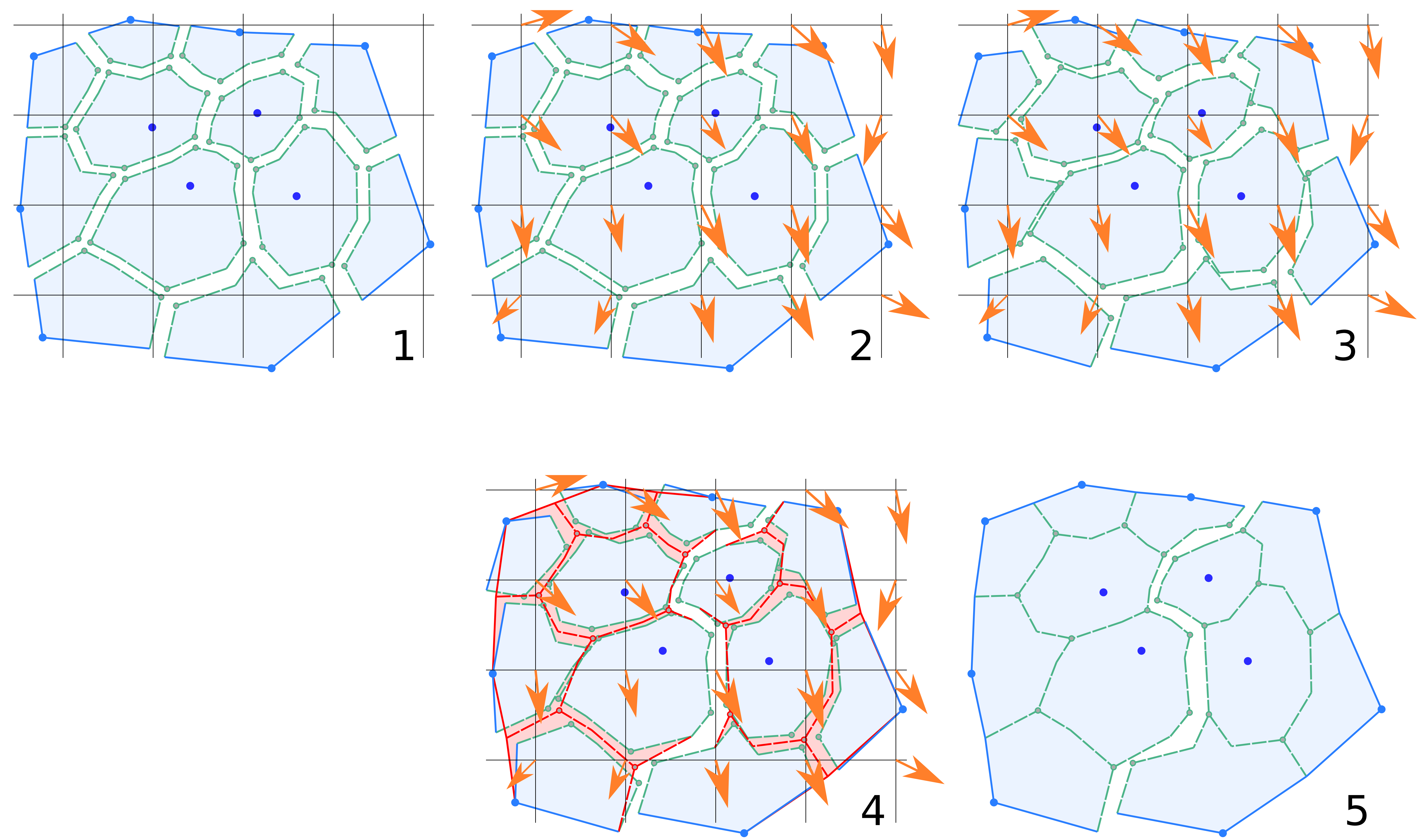


EXTRAPOLATING POSITIONS FOR ADDED VERTICES



EXTRAPOLATING POSITIONS FOR ADDED VERTICES

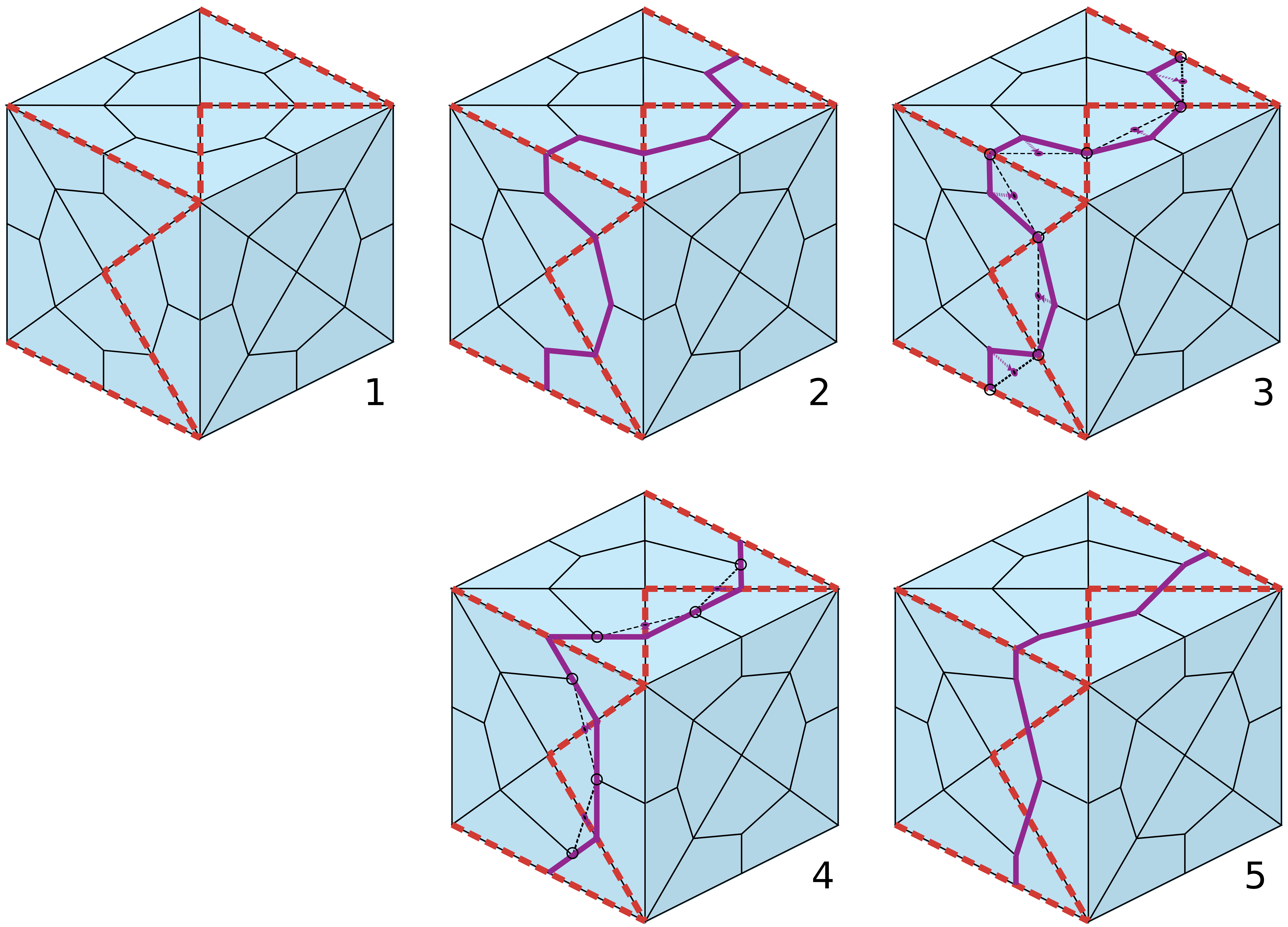
- Granular view
- Locally rigid motion
- Merging vertices based on topology

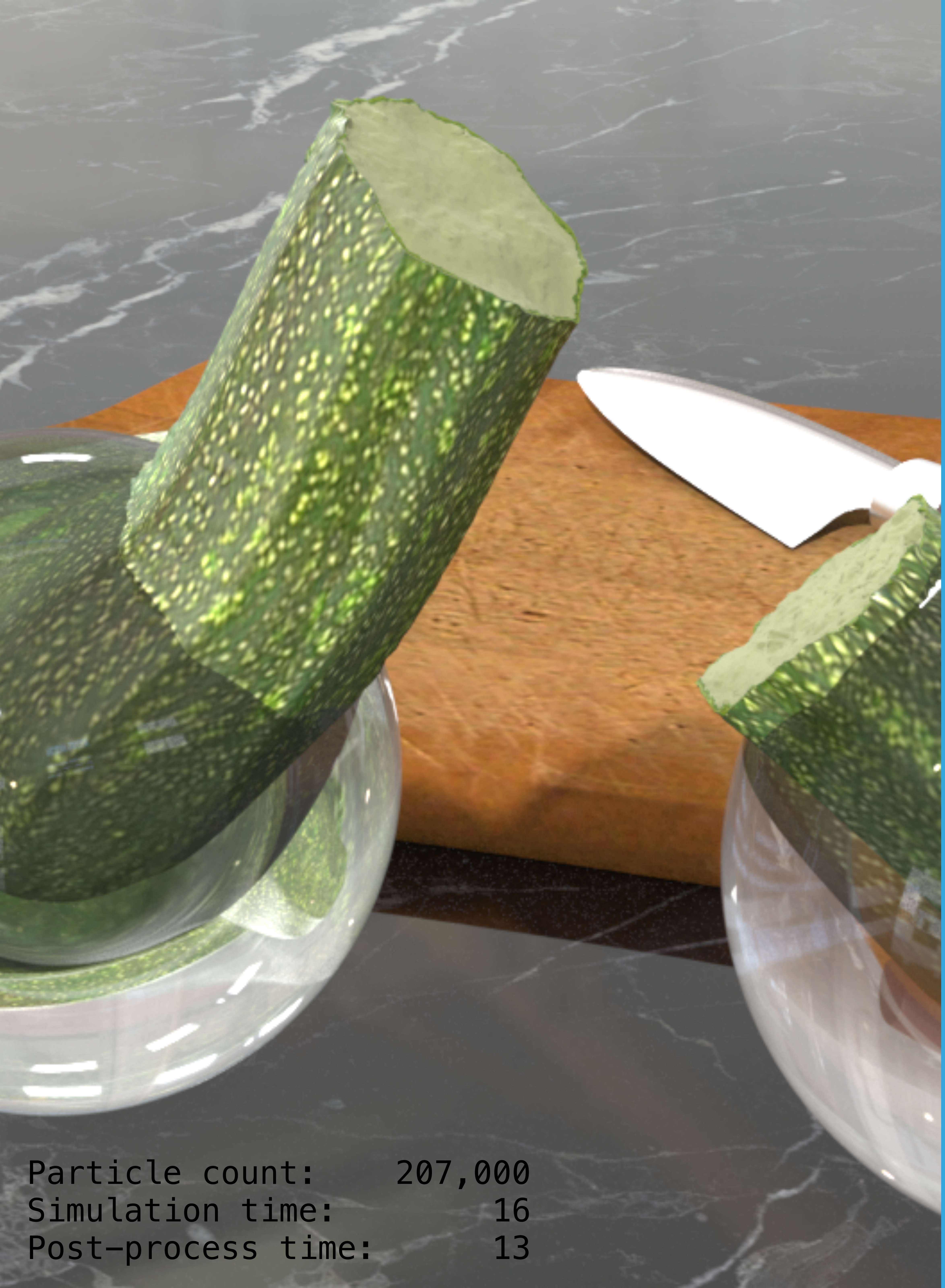


SMOOTHING CRACK SURFACE

SMOOTHING CRACK SURFACE

- Collect all ever broken edges
- Gauss-Siedel smoothing
- Smooth only the undeformed configuration




Particle count: 207,000
Simulation time: 16
Post-process time: 13

DISCUSSION

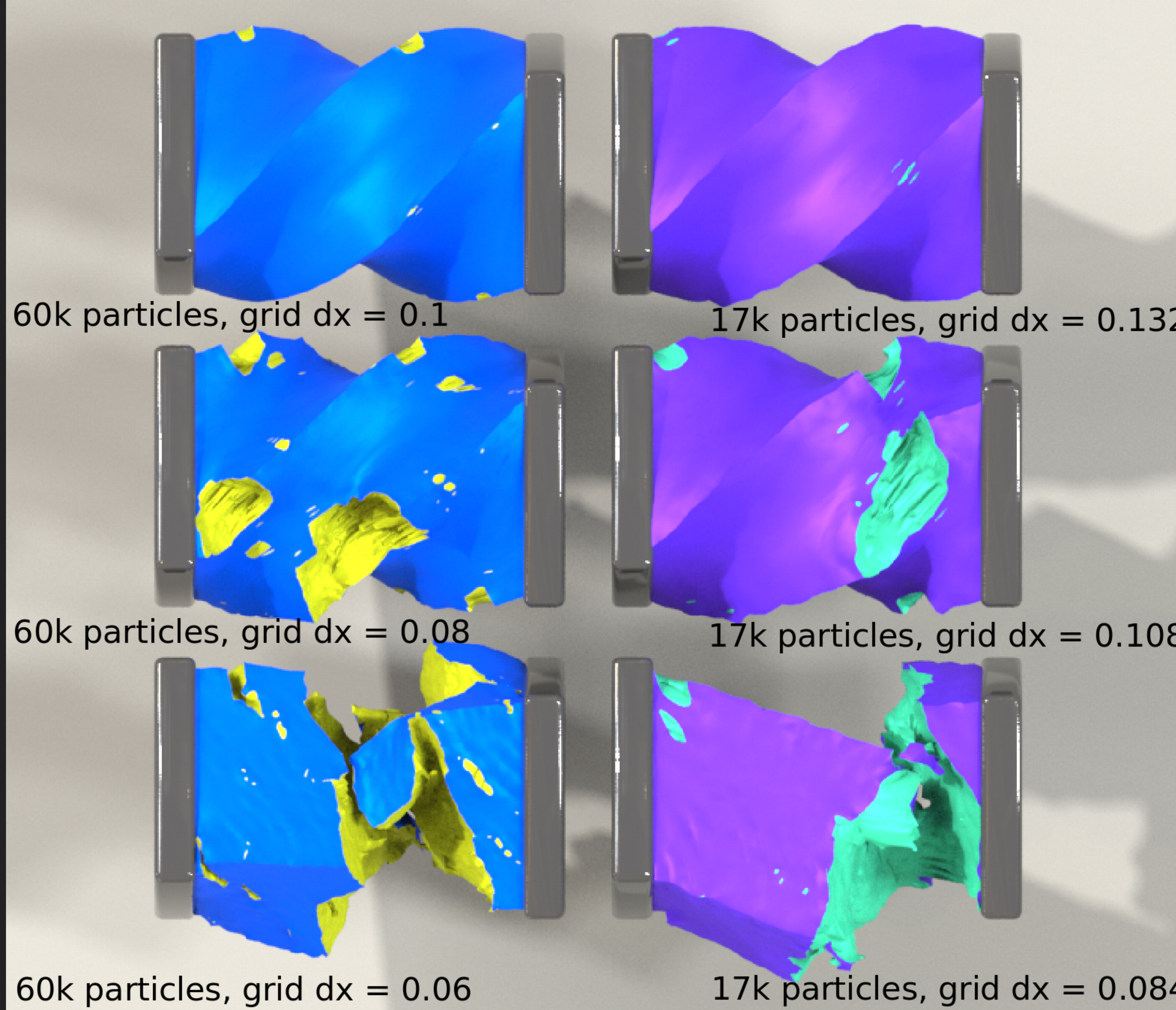
LIMITATIONS AND FUTURE DIRECTIONS

- Crack patterns can be affected by particle sampling density, mesh topology, grid resolution
- Finding appropriate parameters for edge-stretching threshold and crack smoothing iterations
- Exploring different yield surfaces and flow rules

MESH V.S. PARTICLE

Particle-based forces (grid velocity updated F)	Mesh-based forces (mesh geometry updated F)
Delaunay mesh for visualization	requires quality mesh for simulation
has artificial fracture	no artificial fracture
6-8 particles per cell	2 particles per cell
automatic self-collision	
easy coupling with other MPM material	


Small grid dx
Large grid dx
Lagrangian
Particle count: 8,000
Simulation time: 0.6
Post-process time: 0.5


60k particles, grid dx = 0.1
17k particles, grid dx = 0.132
60k particles, grid dx = 0.08
17k particles, grid dx = 0.108
60k particles, grid dx = 0.06
17k particles, grid dx = 0.084


Particle count: 933,000
Simulation time: 50
Post-process time: 5


Particle count: 50,000
Simulation time: 2
Post-process time: 3

ACKNOWLEDGEMENT

- The work is supported by NSF CCF-1422795, ONR (N000141110719, N000141210834), DOD (W81XWH15-1-0147), Intel STC-Visual Computing Grant (20112360) as well as a gift from Adobe Inc.