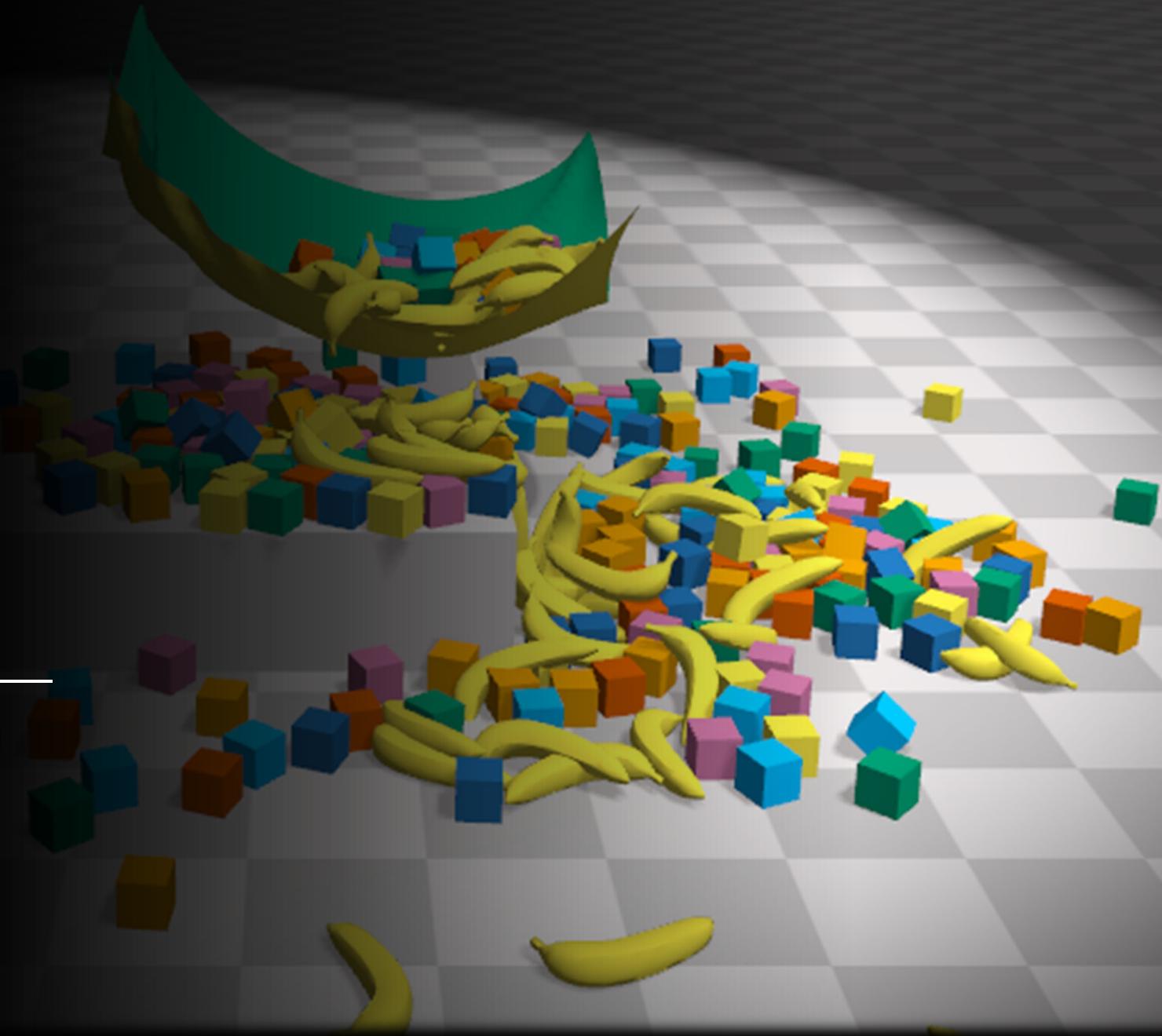


Collision Response

Interior Point Methods

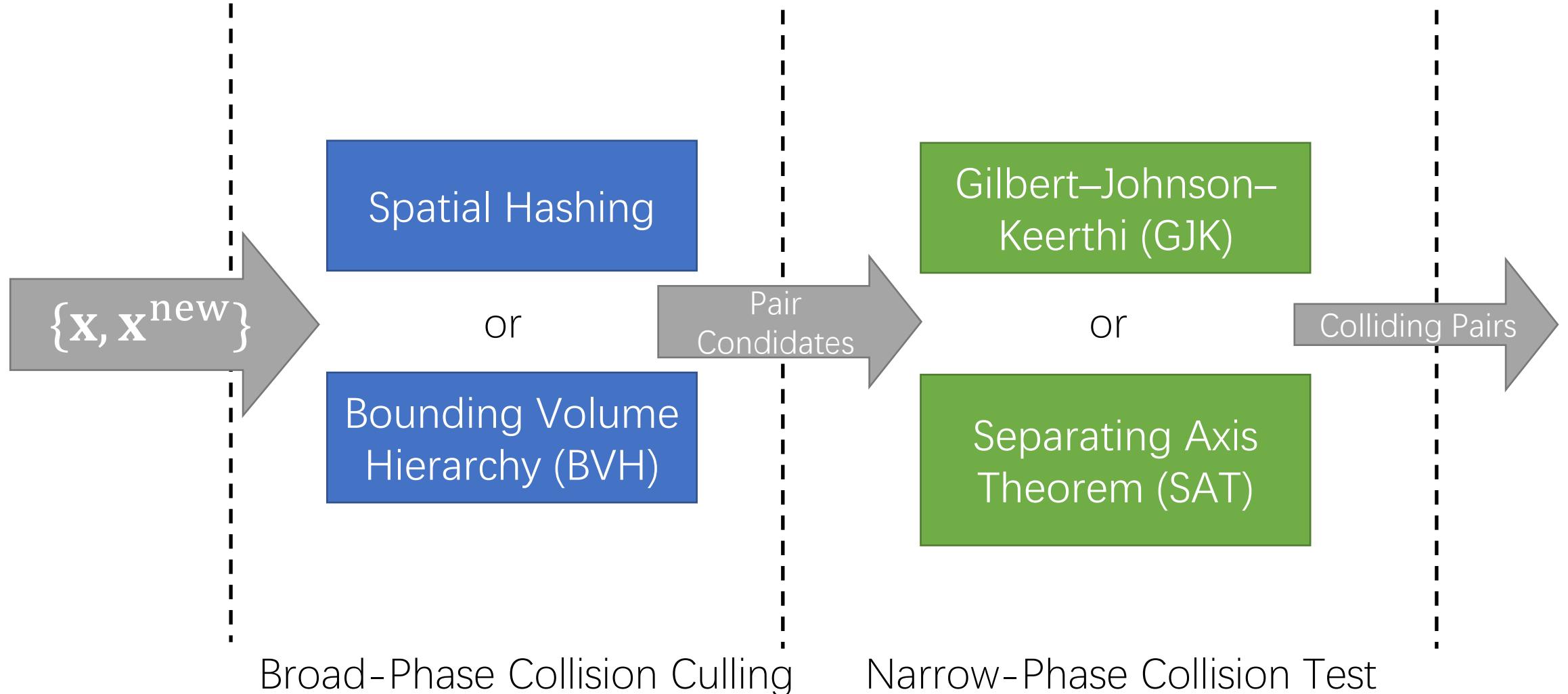
Impact Zone Optimization

Intersection Elimination

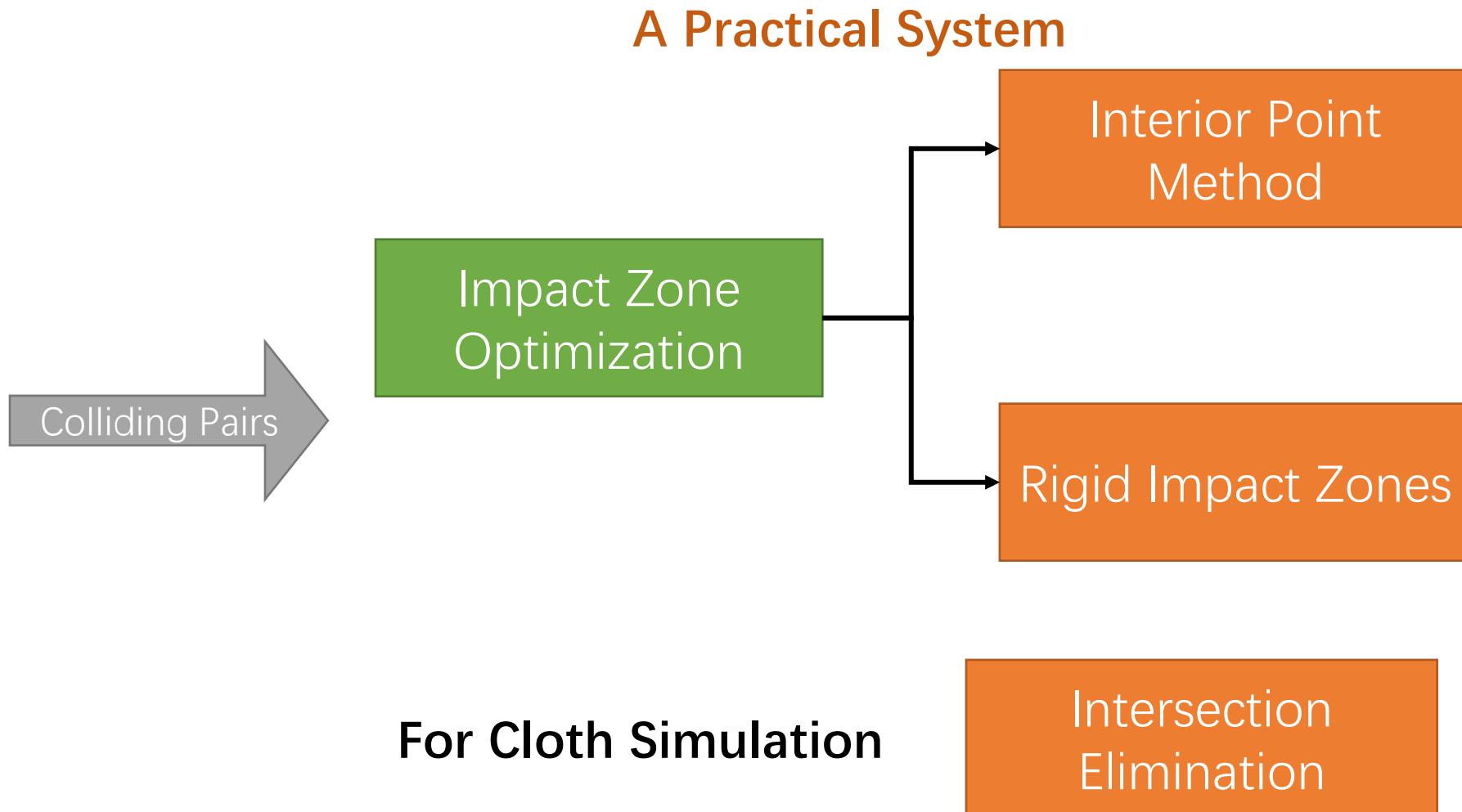


Outline Today

Collision Detection Pipeline



Outline Today



Background – Implicit Simulation

Explicit methods calculate the state of a system at a later time from the state of the system at the current time, while implicit methods find a solution by solving an equation involving both the current state of the system and the later one.

- Implicit Euler: $x^{n+1} = x^n + h\nu^{n+1}, \quad \nu^{n+1} = \nu^n + hM^{-1}f^{n+1}$
- Increment: $\Delta x = h(\nu^n + \Delta\nu), \quad \Delta\nu = hM^{-1}f(x^n + \Delta x, \nu^n + \Delta\nu)$
- 1-st order approximation:

$$\Delta\nu = hM^{-1}(f^n + \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial \nu}\Delta\nu)$$

$$\frac{\partial f}{\partial \nu} \rightarrow -\frac{\partial^2 E}{\partial x \partial \nu}$$
$$\frac{\partial f}{\partial x} \rightarrow -\frac{\partial^2 E}{\partial x^2}$$

- Construct the linear system

$$\left(I - hM^{-1}\frac{\partial f}{\partial \nu} - h^2 M^{-1}\frac{\partial f}{\partial x} \right) \Delta\nu = hM^{-1}(f^n + h \frac{\partial f}{\partial x} \nu^n)$$

Interior Point Method

Given the calculated next state \mathbf{x}^n , we want to update it into $\bar{\mathbf{x}}^{n+1}$, such that the path from \mathbf{x}^n to $\bar{\mathbf{x}}^{n+1}$ is intersection-free.

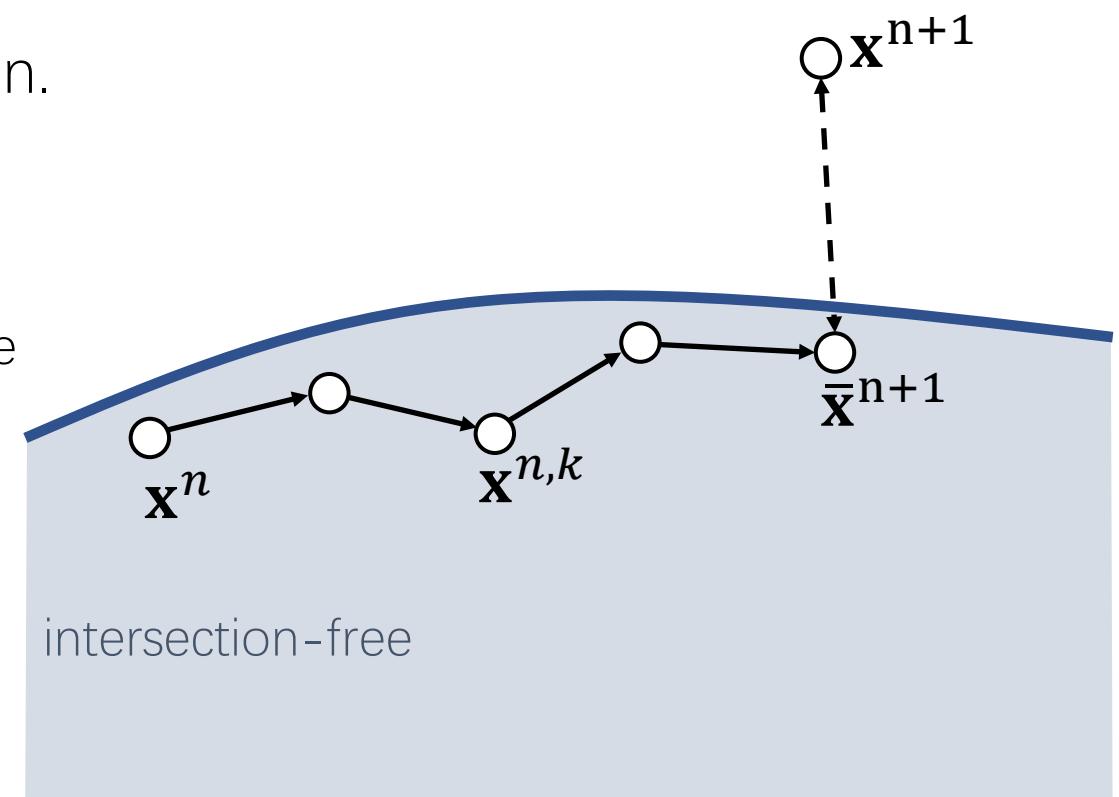
Interior point method reaches a best solution by traversing the interior of the feasible region.

- Do collision detection before assembling linear systems (option)
- Construct collision energies E_{VF}, E_{EE} for all possible collision pairs
- Calculate the Hessian matrix of the energy and assemble the linear system

$$\frac{\partial f}{\partial v} += -\frac{\partial^2 E_{VF}}{\partial v^2} - \frac{\partial^2 E_{EE}}{\partial v^2}$$

$$\frac{\partial f}{\partial x} += -\frac{\partial^2 E_{VF}}{\partial x^2} - \frac{\partial^2 E_{EE}}{\partial x^2}$$

$$f^n += -\frac{\partial E_{VF}}{\partial x} - \frac{\partial E_{EE}}{\partial x}$$



Log-Barrier Interior Point Method

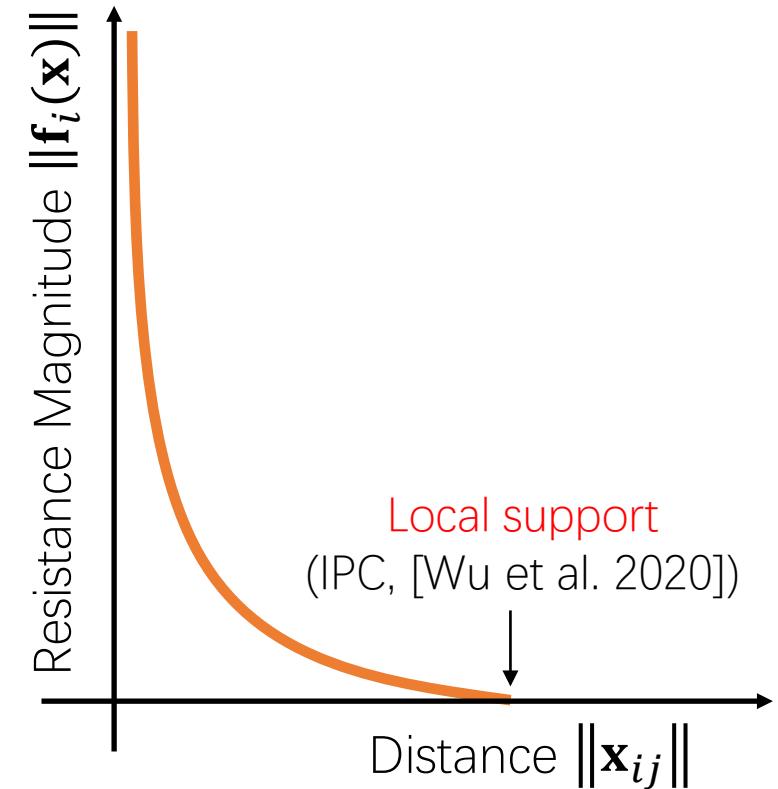
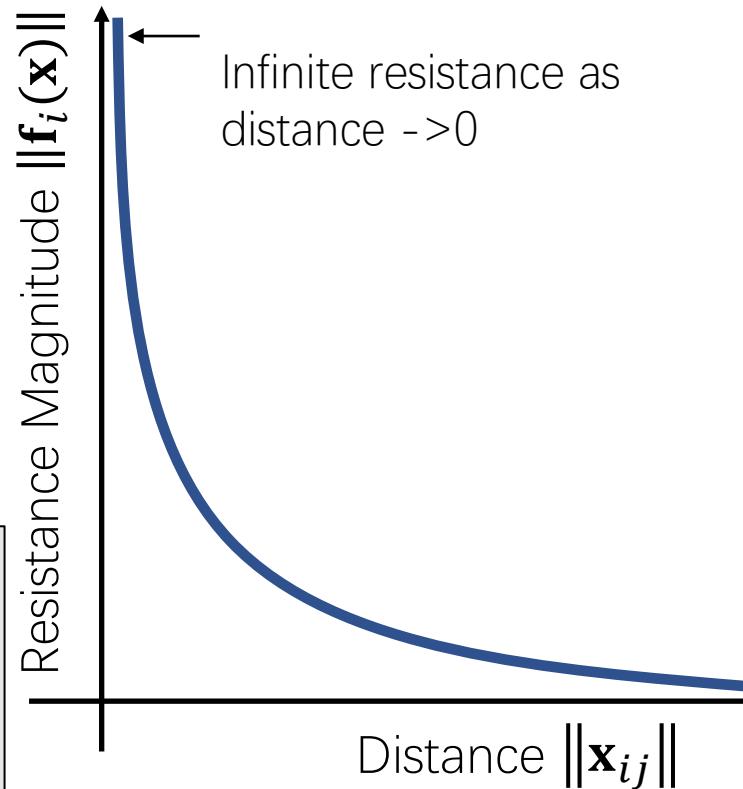
For simplicity, let's consider the Log-barrier repulsion between two vertices.

$$E(\mathbf{x}) = -\rho \log \|\mathbf{x}_{ij}\|$$

$$\mathbf{f}_i(\mathbf{x}) = -\nabla_i E = \rho \frac{\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|^2}$$

$$\mathbf{f}_j(\mathbf{x}) = -\nabla_j E = -\rho \frac{\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|^2}$$

$$E_{IPC}(\mathbf{x}) = \begin{cases} -(\|\mathbf{x}_{ij}\| - \hat{d}) \log \frac{\|\mathbf{x}_{ij}\|}{\hat{d}} \\ 0 \end{cases}$$



Log-Barrier Interior Point Method

We can then formulate the problem as:

$$\bar{\mathbf{x}}^{n+1} \leftarrow \operatorname{argmin}_{\mathbf{x}} \left(\frac{1}{2} \|\mathbf{x} - \mathbf{x}^{n+1}\|^2 - \rho \sum \log \|\mathbf{x}_{ij}\| \right)$$

○ \mathbf{x}^{n+1}

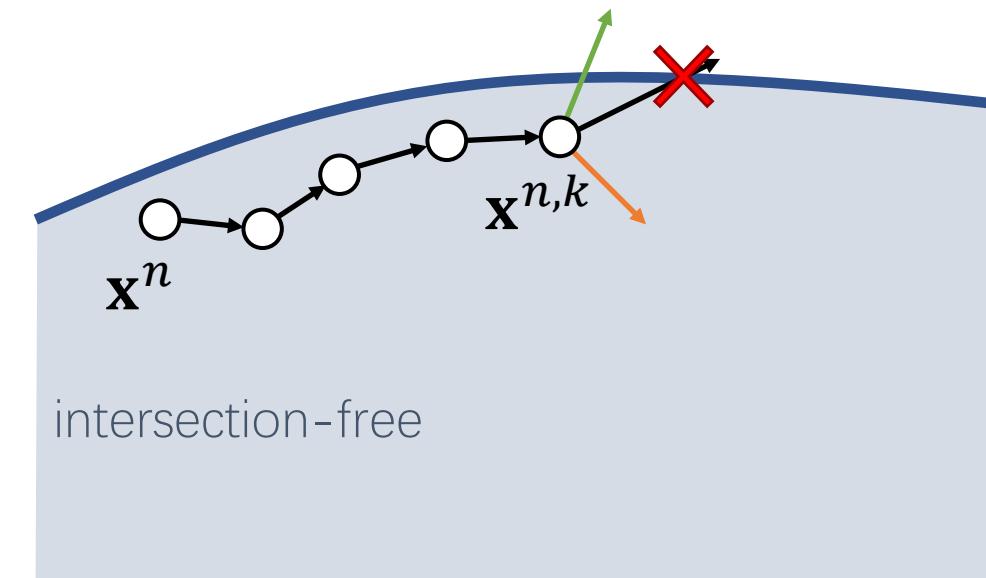
Gradient Descent:

$$\mathbf{x}^{n,0} \leftarrow \mathbf{x}^n$$

For $k = 0 \dots K$

$$\mathbf{x}^{n,k+1} \leftarrow \mathbf{x}^{n,k} + \alpha \left(\mathbf{x}^{n+1} - \mathbf{x}^{n,k} + \rho \sum \frac{\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|^2} \right)$$

$$\bar{\mathbf{x}}^{n+1} \leftarrow \mathbf{x}^{n,k+1}$$



The step size α must be adjusted to ensure that no collision happens on the way. To find α , we need collision tests.

Background – Position Based Dynamics

Position based dynamics (PBD) is based on the projection function.

- The stiffness behavior, i.e., how tightly constraints are enforced, is subject to non-physical factors.
 - The number of iterations
 - The mesh resolution
- The velocity update following projection is important to dynamic effects.
- This method is applicable to other constraints as well, including triangle constraints, volume constraints, and collision constraints.
 - To implement these constraints, simply define their projection functions.

A PBD Simulator

```
//Do Simulation, update  $\mathbf{x}$  and  $\mathbf{v}$ 
 $\mathbf{v}^{n+1} \leftarrow \alpha(\mathbf{v}^n + hM^{-1}f_{ext})$ 
 $\mathbf{x}^{n+1} \leftarrow \mathbf{x}^n + \mathbf{v}^{n+1}$ 
//Now PBD starts.
 $\mathbf{x}^{\text{new}} \leftarrow \text{Projection}(\mathbf{x}^{n+1})$ 
 $\mathbf{v}^{n+1} \leftarrow \mathbf{v}^{n+1} + (\mathbf{x}^{\text{new}} - \mathbf{x}^{n+1})/h$ 
 $\mathbf{x}^{n+1} \leftarrow \mathbf{x}^{\text{new}}$ 
```

Impact Zone Optimization

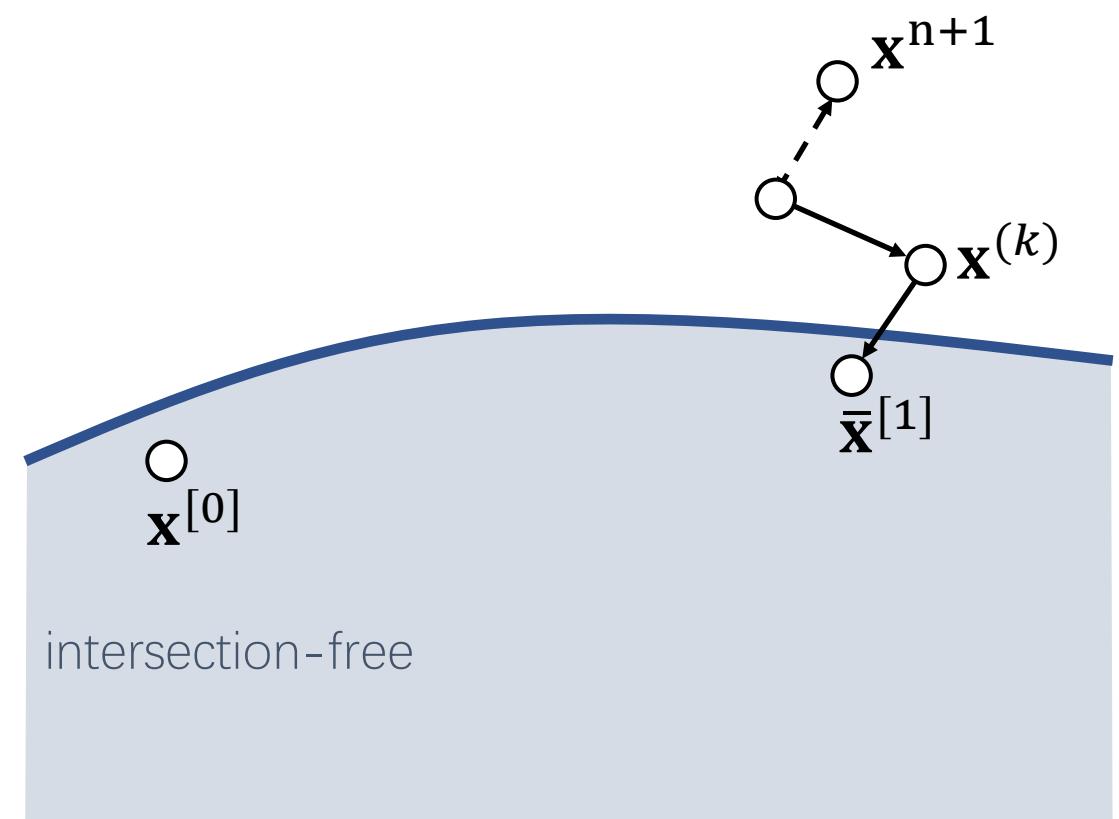
Given the calculated next state \mathbf{x}^n , we want to update it into $\bar{\mathbf{x}}^{n+1}$, such that the path from \mathbf{x}^n to $\bar{\mathbf{x}}^{n+1}$ is intersection-free.

The goal of impact zone optimization is to optimize \mathbf{x}^{n+1} until it becomes intersection-free.

A PBD Simulator

```
//Do Simulation, update  $\mathbf{x}$  and  $\mathbf{v}$ 
 $\mathbf{v}^{n+1} \leftarrow \alpha(\mathbf{v}^n + hM^{-1}f_{ext})$ 
 $\mathbf{x}^{n+1} \leftarrow \mathbf{x}^n + \mathbf{v}^{n+1}$ 
//Now PBD starts.
 $\mathbf{x}^{new} \leftarrow \text{Projection}(\mathbf{x}^{n+1})$ 
 $\mathbf{v}^{n+1} \leftarrow \mathbf{v}^{n+1} + (\mathbf{x}^{new} - \mathbf{x}^{n+1})/h$ 
 $\mathbf{x}^{n+1} \leftarrow \mathbf{x}^{new}$ 
```

Collision
Detection

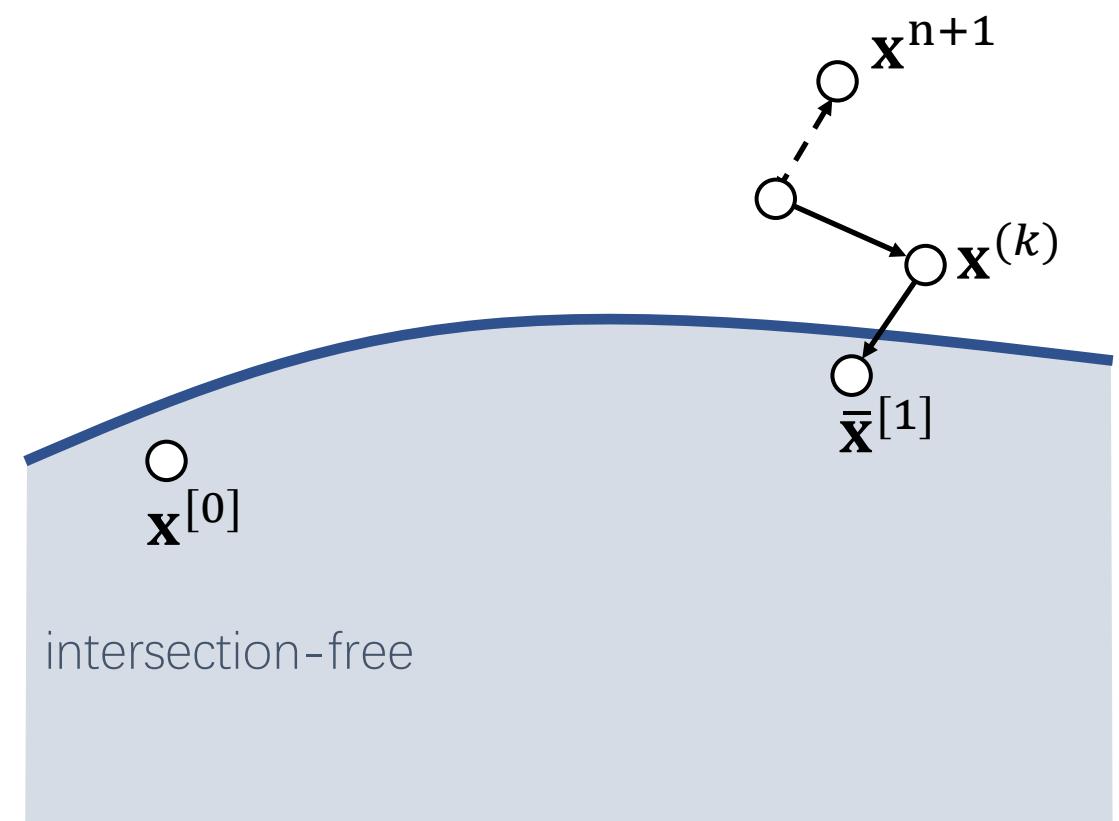


Impact Zone Optimization

Given the calculated next state \mathbf{x}^n , we want to update it into $\bar{\mathbf{x}}^{n+1}$, such that the path from \mathbf{x}^n to $\bar{\mathbf{x}}^{n+1}$ is intersection-free.

The goal of impact zone optimization is to optimize \mathbf{x}^{n+1} until it becomes intersection-free.

- Vertex-face pair: vertex-face distance constraints
- Edge-edge pair: edge-edge distance constraints
- Special case: Capsule constraints, etc.



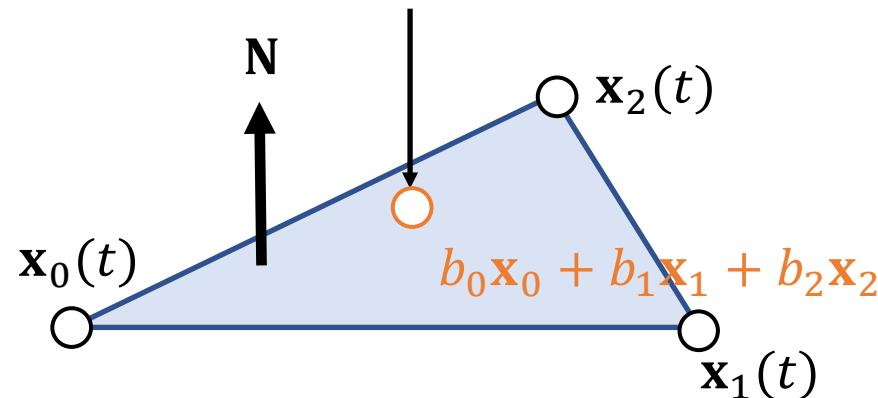
Impact Zone Optimization

The goal of impact zone optimization is to optimize \mathbf{x}^{n+1} until it becomes intersection-free.

$$C(\mathbf{x}) = (\mathbf{x}_3 - b_0\mathbf{x}_0 - b_1\mathbf{x}_1 - b_2\mathbf{x}_2) \cdot \mathbf{N} \geq 0$$

$$\frac{\partial C}{\partial \mathbf{x}_i} = -b_i \mathbf{N}, \quad i = 0, 1, 2$$

$$\frac{\partial C}{\partial \mathbf{x}_3} = \mathbf{N}$$

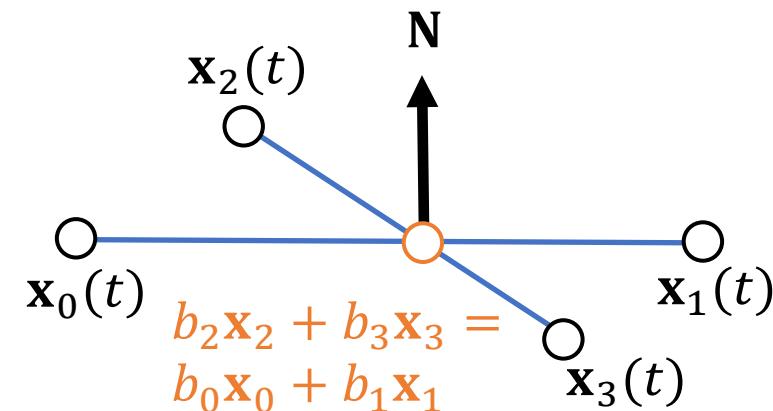


Vertex-Face Pair

$$C(\mathbf{x}) = (b_2\mathbf{x}_2 + b_3\mathbf{x}_3 - b_0\mathbf{x}_0 - b_1\mathbf{x}_1) \cdot \mathbf{N} \geq 0$$

$$\frac{\partial C}{\partial \mathbf{x}_i} = -b_i \mathbf{N}, \quad i = 0, 1$$

$$\frac{\partial C}{\partial \mathbf{x}_j} = b_j \mathbf{N}, \quad j = 2, 3$$



Edge-Edge Pair

Impact Zone Optimization

The goal of impact zone optimization is to optimize \mathbf{x}^{n+1} until it becomes intersection-free.

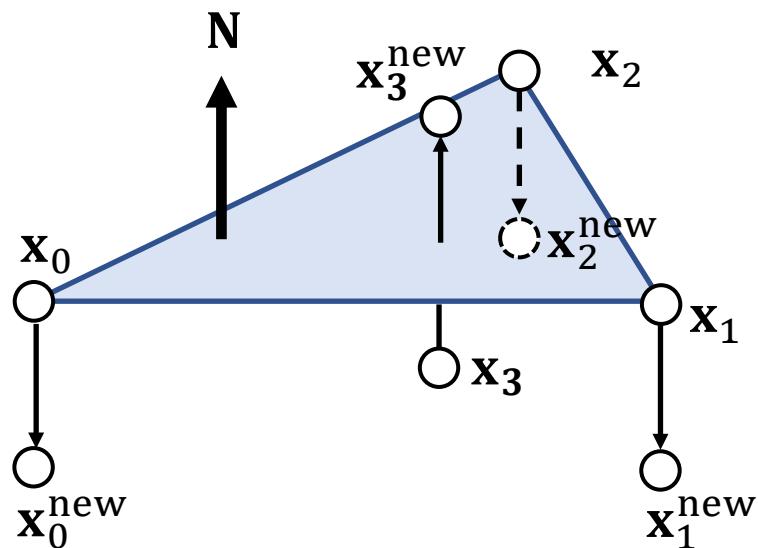
$$C(\mathbf{x} + \Delta\mathbf{x}) = 0$$

$$C(\mathbf{x}) + \nabla C(\mathbf{x})\Delta\mathbf{x} = 0$$

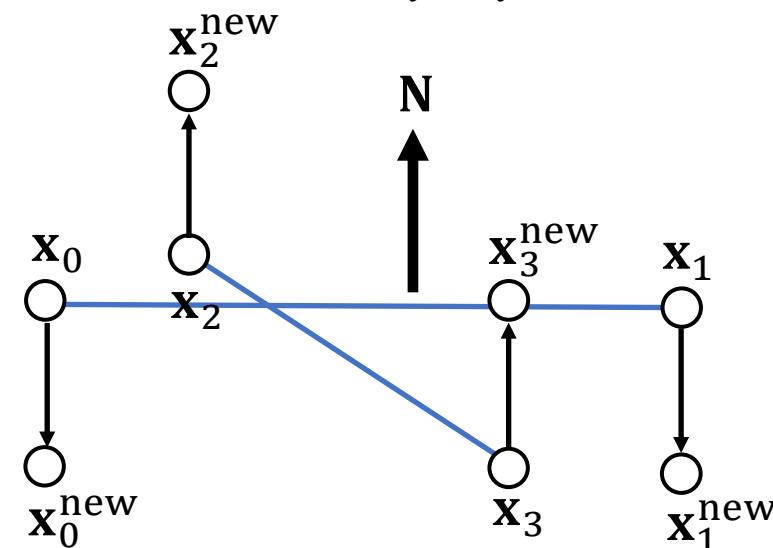
$$\Delta\mathbf{x} = \lambda \nabla C(\mathbf{x})$$

$$\lambda = -\frac{C(\mathbf{x})}{|\nabla C(\mathbf{x})|^2}$$

$$\Delta\mathbf{x}_i = \frac{\lambda}{m_i} \frac{\partial C}{\partial \mathbf{x}_i}$$



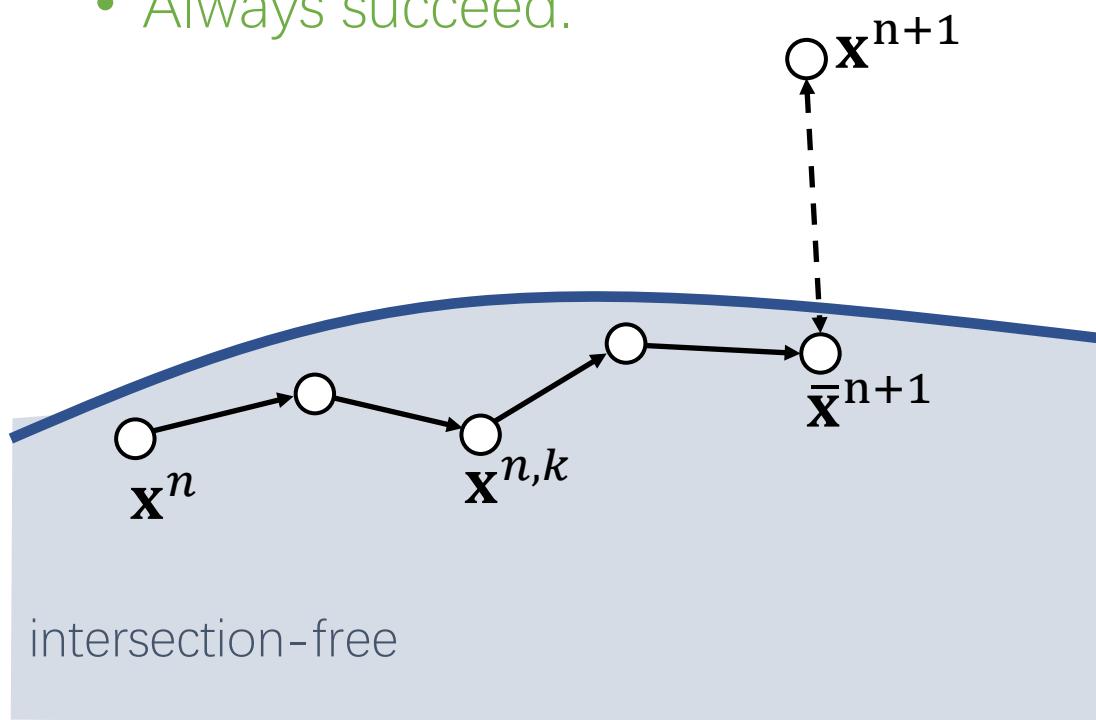
Vertex-Face Pair



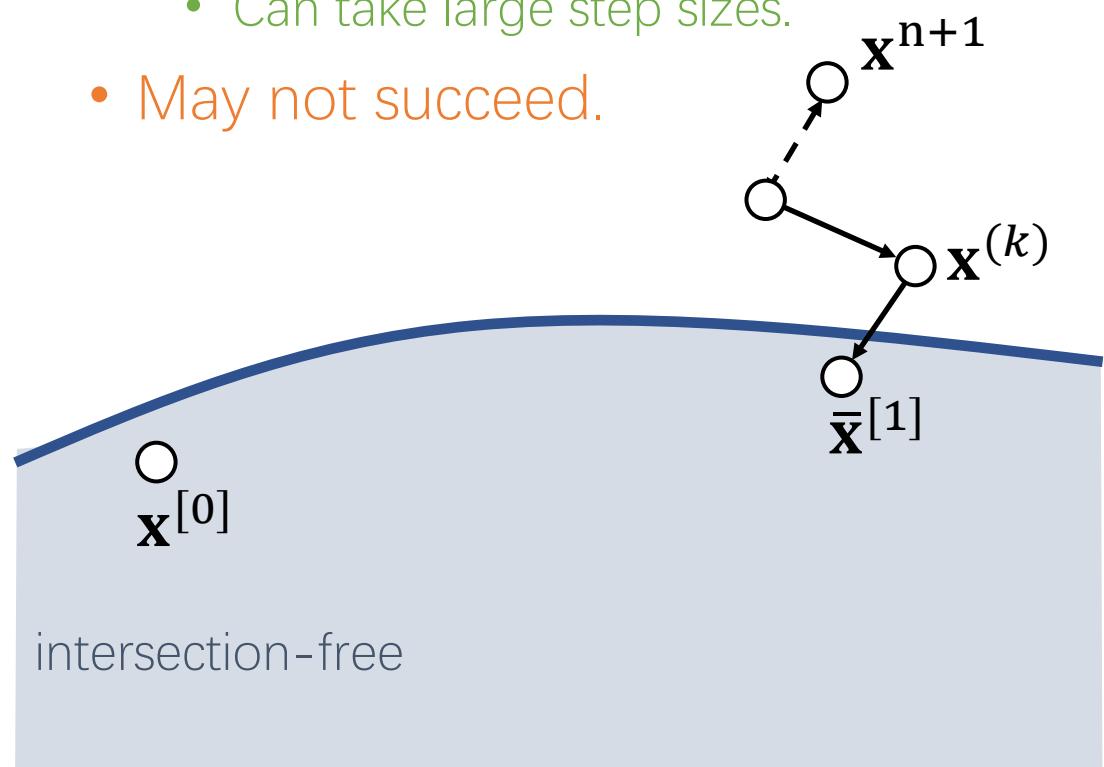
Edge-Edge Pair

Pros and Cons

- Slow.
 - Far from solution.
 - All of the vertices.
 - Cautiously by small step sizes.
- Always succeed.

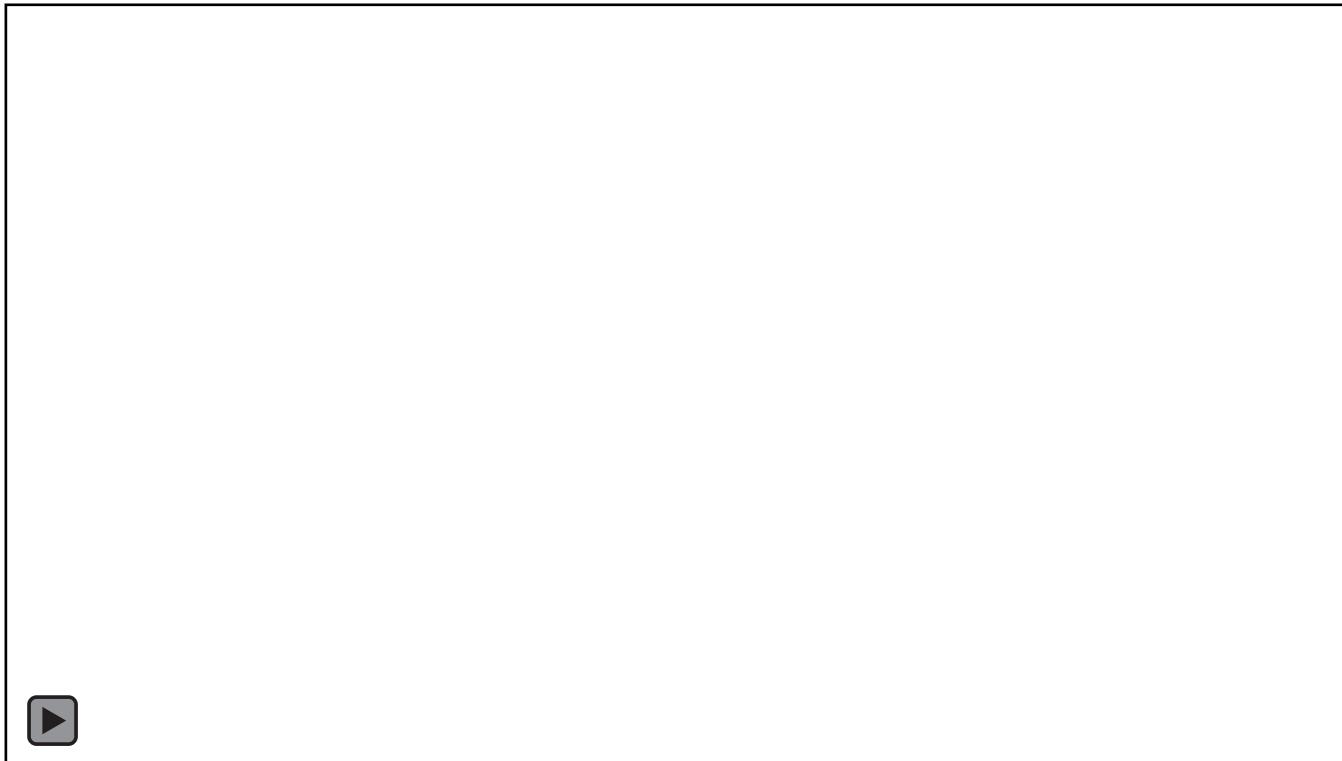


- Fast.
 - Close to solution.
 - Only vertices in collision (impact zones).
 - Can take large step sizes.
- May not succeed.

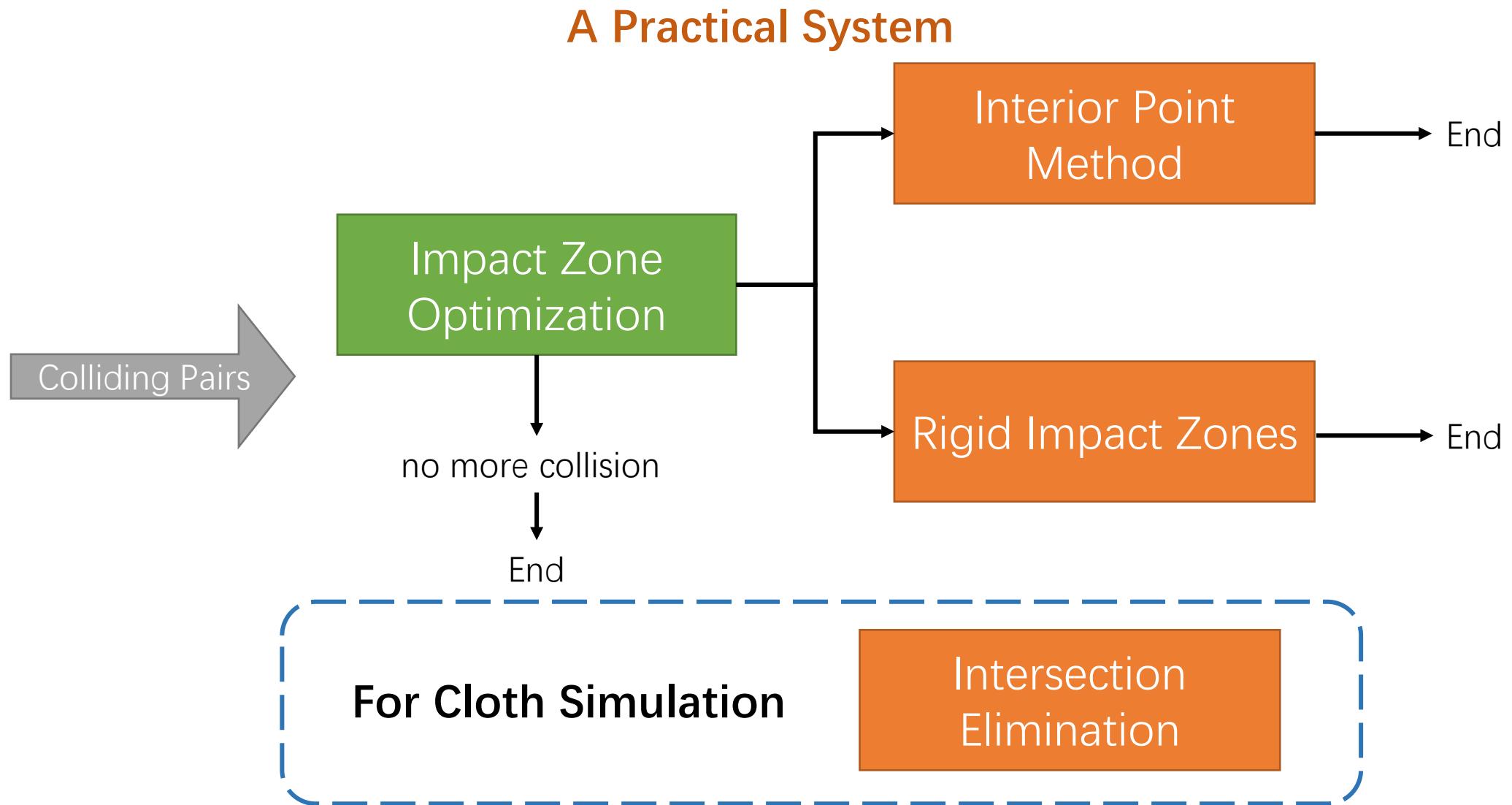


Rigid Impact Zones

The rigid impact zone method simply freezes vertices in collision from moving in their pre-collision state. It's simple and safe, but has noticeable artifacts.



Outline Today

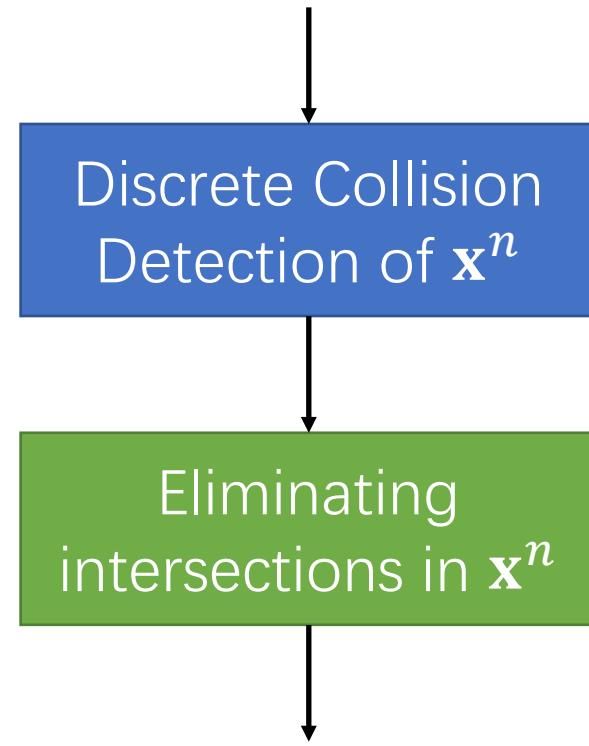


Intersection Elimination



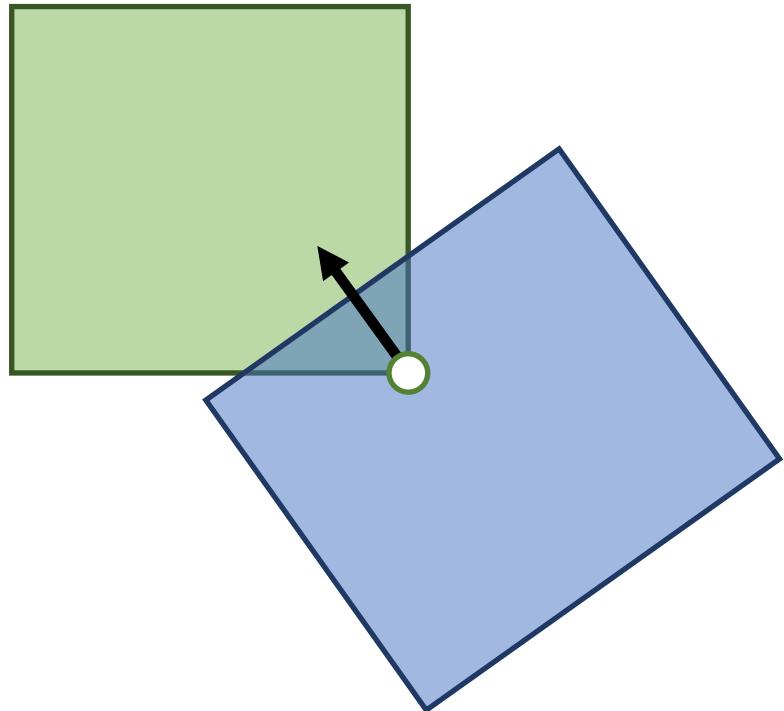
Intersection Elimination

- Let's consider how to eliminate existing intersections, but without using any collision history.
- Such a method is useful when there are already intersections in simulation, due to:
 - Past collision handling failures
 - Intense user interaction
- In this case, we don't require the simulation is to always intersection-free.

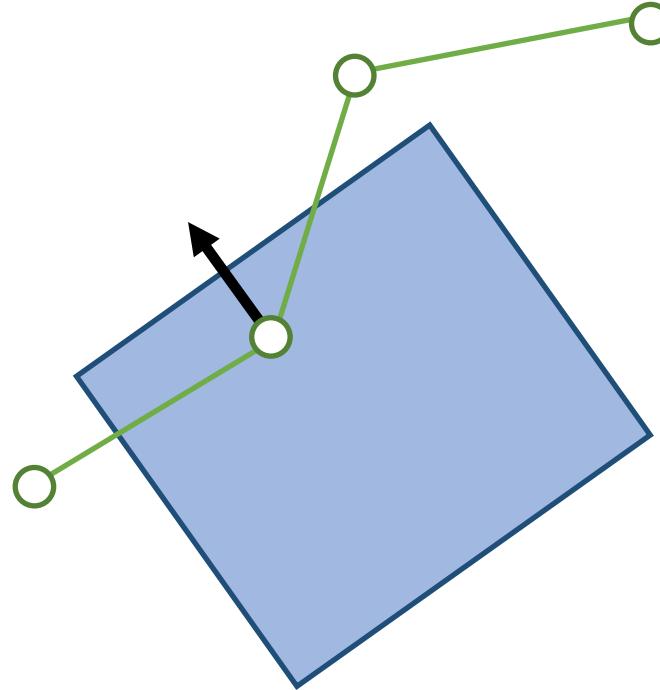


Intersection Elimination

Eliminating cloth-volume and volume-volume intersections is straightforward: simply pushing vertices/edges in the volume out.



Volume-volume intersection

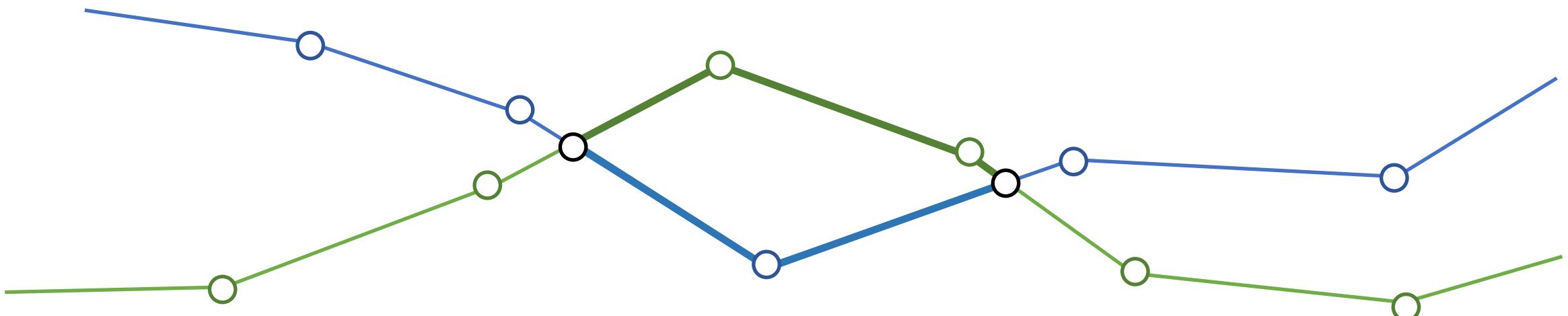


Cloth-volume intersection

Intersection Elimination

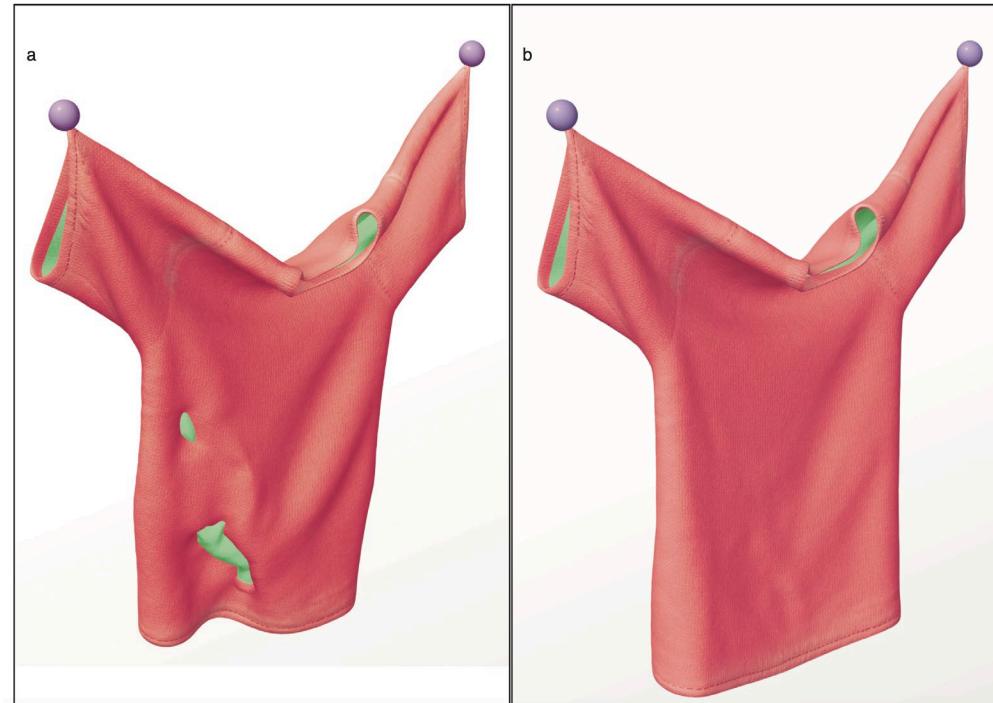
The situation is complicated in cloth-cloth intersection, since we don't have a clear definition of inside and outside.

Baraff et al. used flood-fill to segment cloth into regions and decided which region is in intersection. (**Cannot handle boundary well.**)



Intersection Elimination

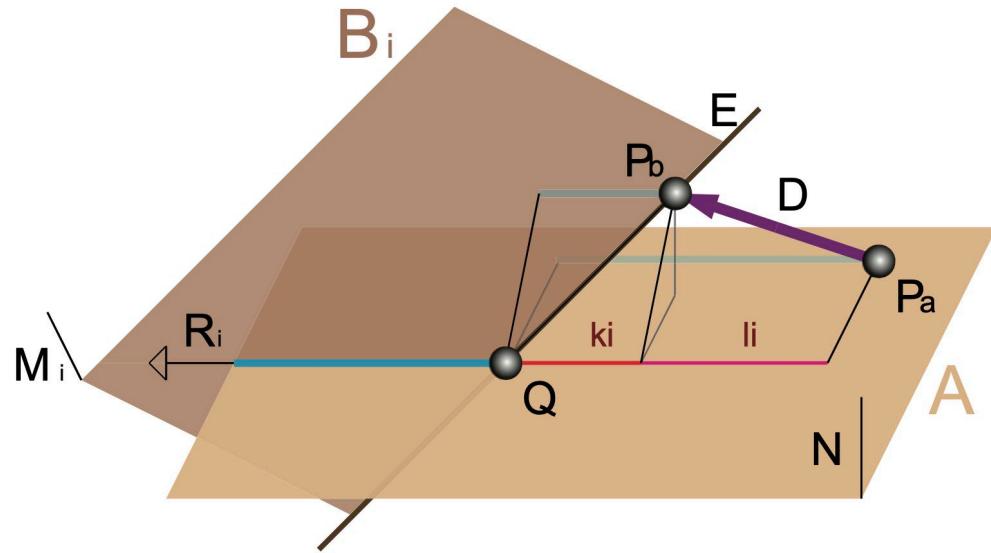
Baraff et al. used flood-fill to segment cloth into regions and decided which region is in intersection. (Cannot handle boundary well.)



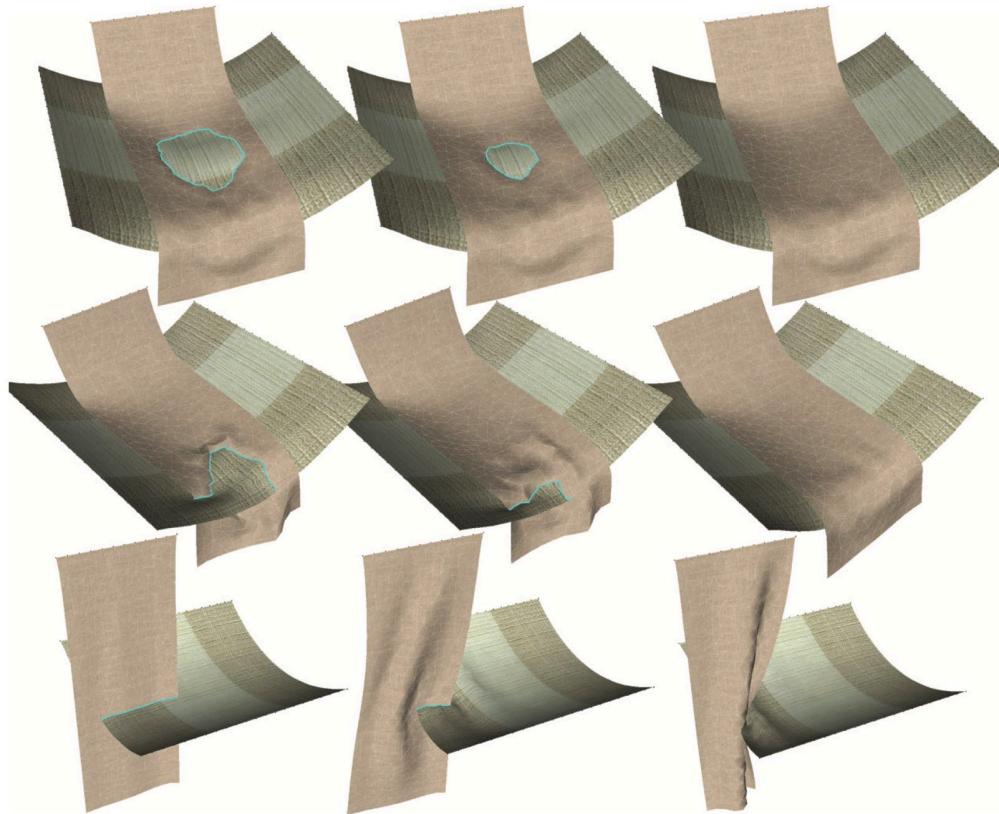
Intersection Elimination

Volino and Magnenat-Thalmann proposed to untangle cloth by reducing the intersection contour.

Their method can handle boundaries, but it doesn't always work.



$$G = \sum_i \left(R_i - 2 \frac{E \cdot R_i}{E \cdot N} N \right)$$



Reference and Further Readings

- GAMES103-基于物理的计算机动画入门 [[Link](#)]
- Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics [[Link](#)]
- Position Based Dynamics [[Link](#)]
- Untangling Cloth [[Link](#)]
- Resolving surface collisions through intersection contour minimization [[Link](#)]

All relevant files (ppt, pdf, image, code, video):

https://github.com/LIKOTYO/Lab-Presentation/tree/master/20230522_Collision_Handling