MGPBD: A Multigrid Accelerated Global XPBD Solver

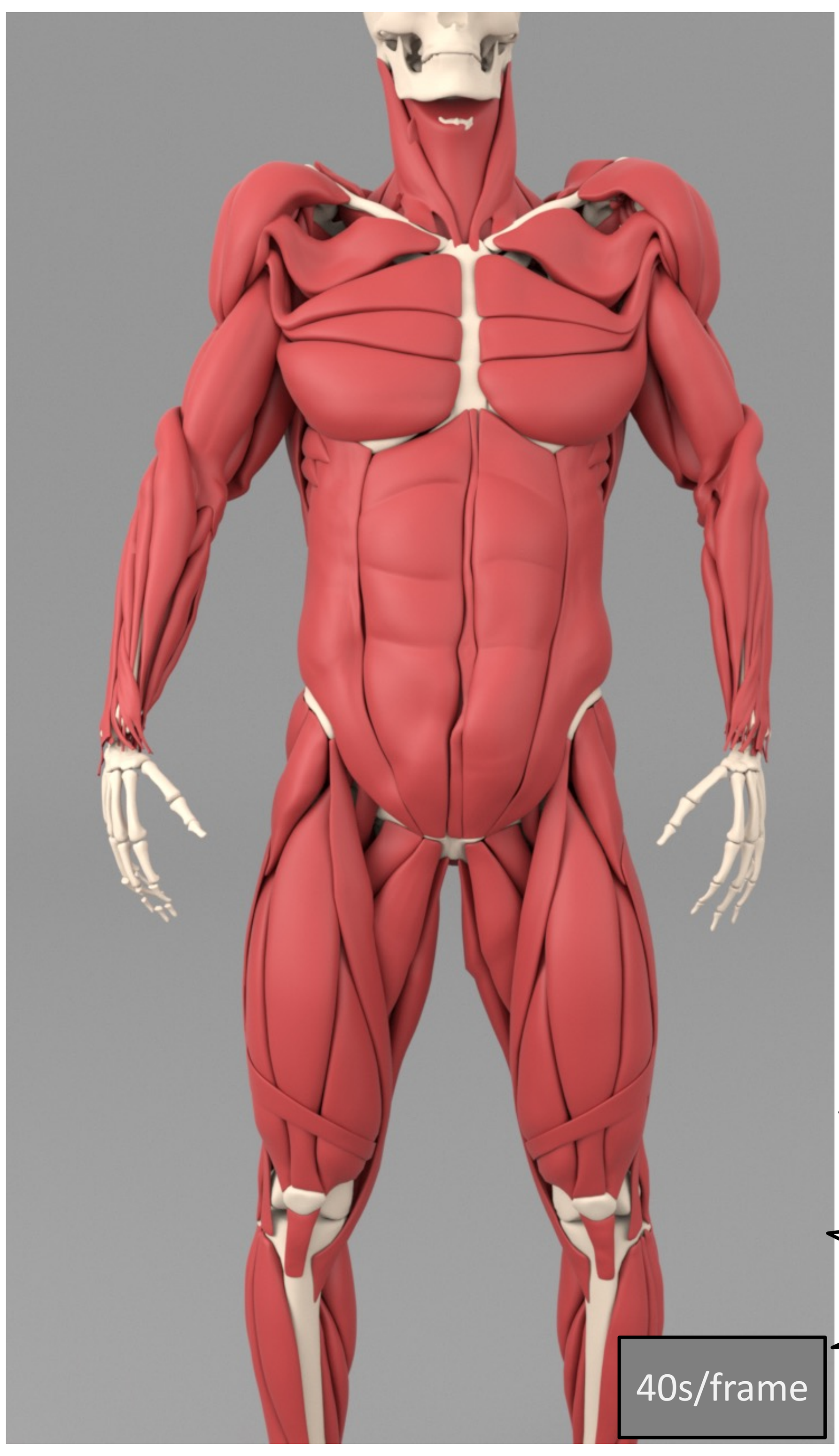
Chunlei Li[1*], Peng Yu[1*], Tiantian Liu[2], Siyuan Yu[3], Yuting Xiao[1], Shuai Li[1†], Aimin Hao[1], Yang Gao[1†], Qinping Zhao[1]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, China
[2]Taichi Graphics, China.
[3]Zenustech, China.
*Both authors contributed equally to this research
†corresponding authors: *lishuai@buaa.edu.cn, gaoyangvr@buaa.edu.cn*

https://github.com/chunleili/mgpbd

arXiv https://arxiv.org/abs/2505.13390

Code & Paper

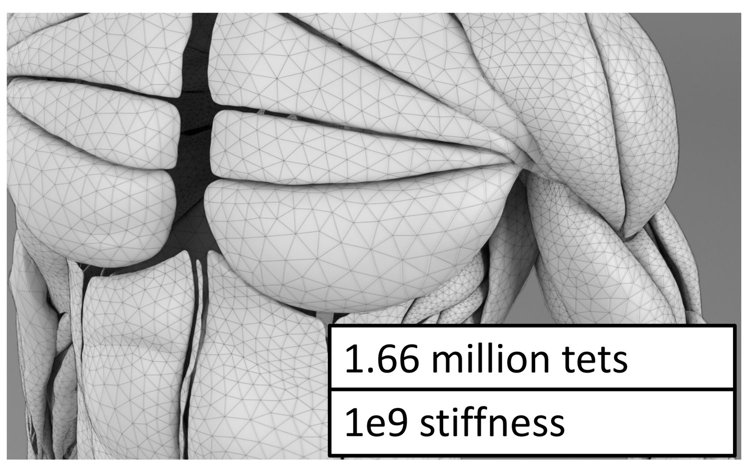


Motivation & Background

Muscle Simulation

High resolution: 1.66 million tets

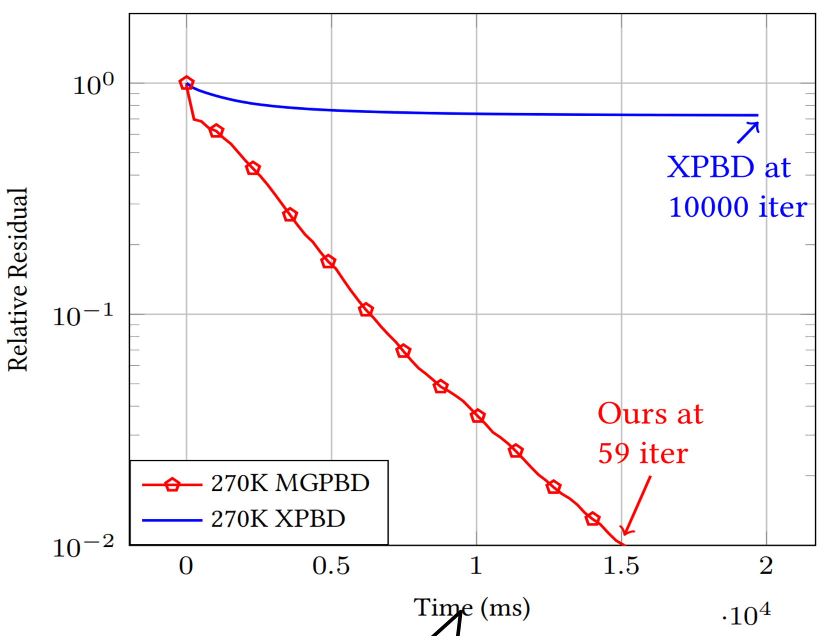
High stiffness: hard like car tires (G=10MPa)



XPBD[1]: 1) Universal 2) Simple 3)Fast

One step of PBD/XPBD

1. Move under inertia
2. Solve constraints
3. Update velocities



XPBD[1]	SAP[3]	Newton	Ours
Dual space	Dual Space	Primal Space	Dual Space
Local system	Global system	Global system	Global system
Non-linear GS/Jacobi	Direct Solver	All types of linear solvers	AMG

XPBD does NOT converge even using **10,000** iterations!

- The higher mesh **resolution**, the more difficult
- The higher **stiffness**, the more difficult

XPBD

- Performs well in low-res. low stiffness simulations
- Performs NOT well in high res. high stiffness simulations

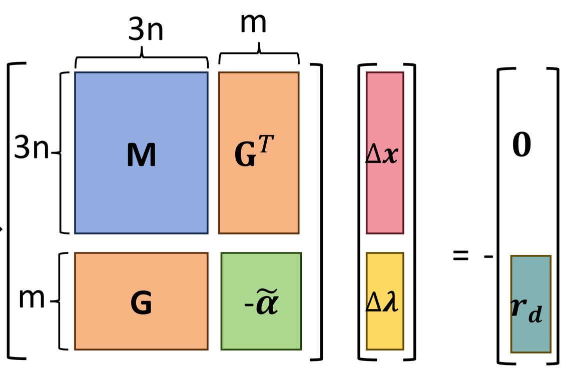
Why XPBD's linear system has **stalling** issue?

Method

Dual Space Global System

For simplicity $G \coloneqq \nabla C(x) \in R^{m \times 3n}$

KKT system $\begin{bmatrix} \mathbf{M} & \mathbf{G}^T \\ \mathbf{G} & \tilde{\alpha} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = -\begin{bmatrix} \mathbf{0} \\ \mathbf{C} + \tilde{\alpha}\lambda \end{bmatrix}$



Take Shur complement, get our system to solve

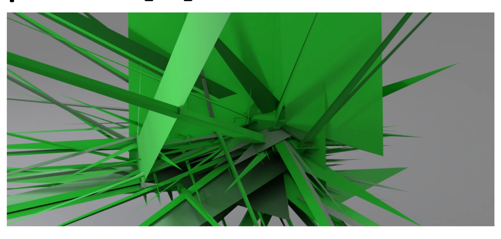
$$\underbrace{(\mathbf{GM}^{-1}\mathbf{G}^T + \tilde{\alpha})}_{A}\,\Delta\lambda = \underbrace{-\mathbf{C} - \tilde{\alpha}\lambda}_{b}$$

Dual System: DOF is $\Delta\lambda$

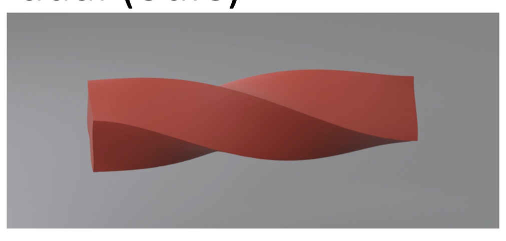
$\Delta\lambda_j = b_j / A_{jj}$ for $j = 1,2,\ldots,m$

	Dual Space	Primal Space[2]
DOF	$\Delta\lambda$	Δx
Size	mxm	3nx3n
System	$GM^{-1}G^T + \alpha/\Delta t^2$	$G^T\alpha^{-1}G + M/\Delta t^2$
Pros	Stable at high stiffness ratio	Stable at high mass ratio
Cons	Unstable at high mass ratio	Unstable at high stiffness ratio

primal[4]



dual (ours)



XPBD: Non-linear GS/Jaocbi

- Local System: Ignores the off-diag terms
- One Sweep: one sweep to solve the linear system.

VS.

MGPBD: Dual Space + Global Solver + AMG

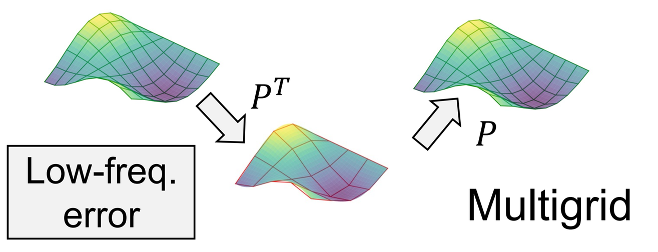
- Global System: solve the full system with AMG
- Lazy setup: reuse P between steps.
- Use near kernel $K = \{x | Ax \approx 0\}$ to approximate the coarse space

Algebraic Multigrid

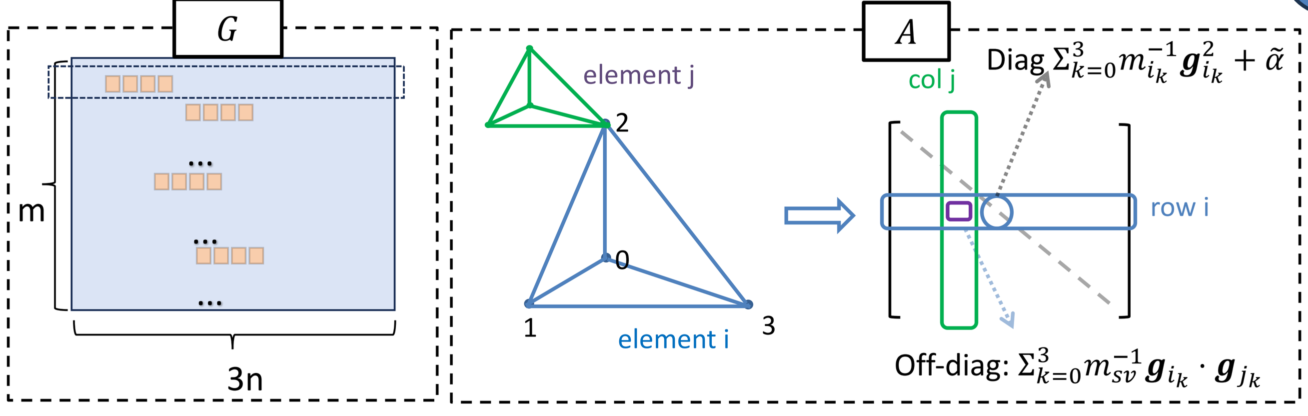
Iterative method: Only eliminates high-frequency errors, hardly **low-frequency** ones.

Multigrid method: Transfer errors to coarse grids, where **low-frequency** errors become high-frequency.

$$A = GM^{-1}G^T + \alpha/\Delta t^2$$



- GMG(Geometric Multigrid): hard to find coarse-fine relations in the dual space
- **AMG**(Algebraic Multigrid): only based on matrix A, but building P cost **2/3** of total time

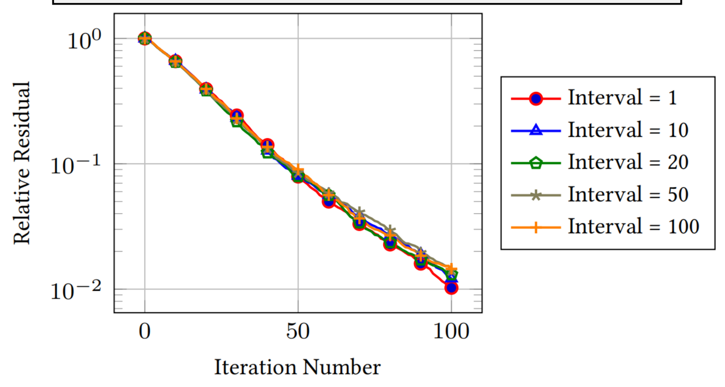


Diag: $\Sigma_{k=0}^{3} m_{ik}^{-1} g_{ik}^2 + \tilde{\alpha}$

Off-diag: $\Sigma_{k=0}^{3} m_{sv}^{-1} g_{ik} \cdot g_{jk}$

A is an adjacent graph → AMG: Building P only relies on A → Can we reuse the P? → Yes!

1. **Lazy setup**: minor convergence loss, save 2/3 of time!



K: Near kernel $K: \{x | Ax \approx 0\}$ —estimate→ coarse space —approx.→ L: The space **low-freq.** errors reside (Goal)

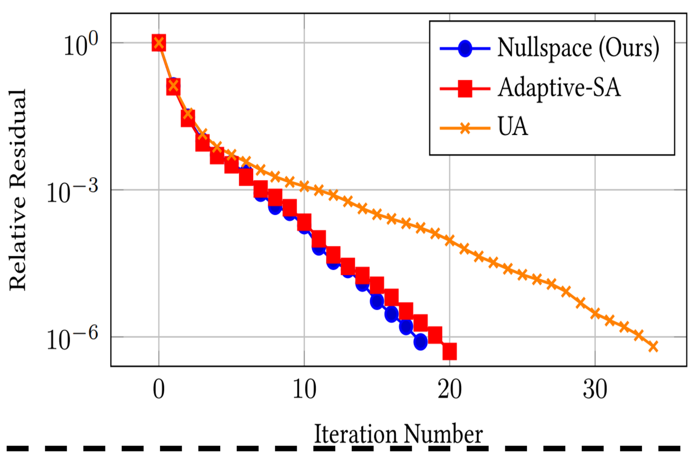
coarse space —span— P's **columns** as **basis**

$Pe_c = e_f$

Heuristic rule of AMG: Space L: $Ax \approx 0$

sufficient smoothing → Low-freq left, $Ax \approx 0$

2. **Near Kernel: Doubled** the convergence speed!

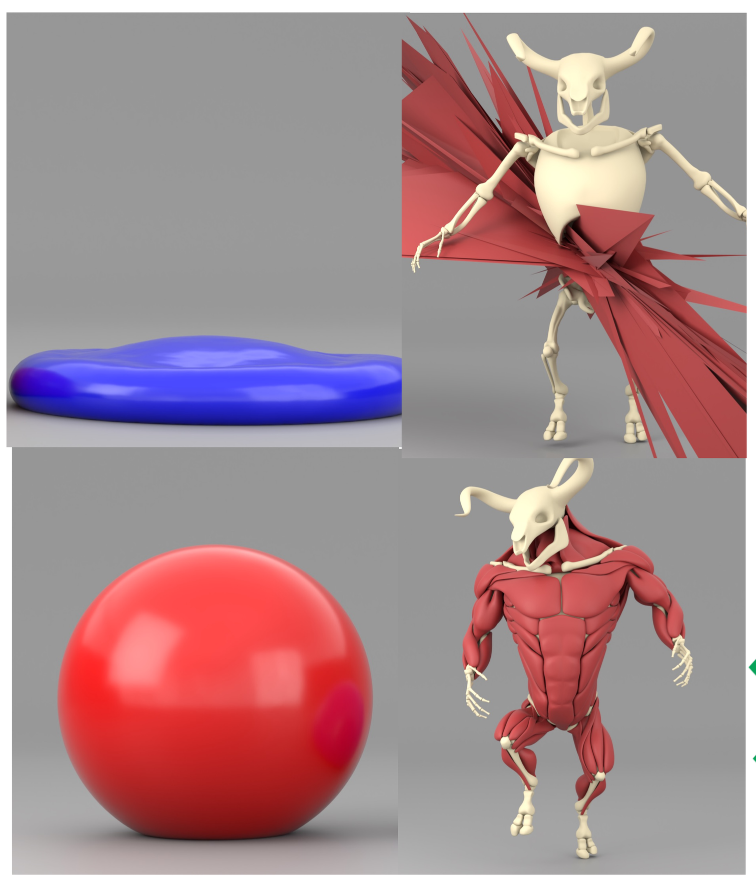


Algorithm

Algorithm 1 MGPBD Simulation Loop

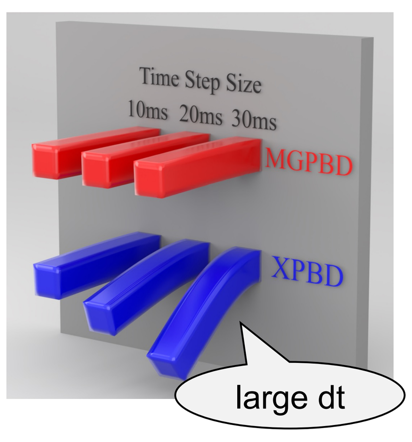
```
x̃,x,x_old, v ← semiEuler(v,Δt,f_ext)
λ ← (0, ..., 0)^T
for ite = 0, 1, ..., maxiter do
    calculate C and ∇C
    assemble A ← ∇CM^{-1}∇C^T + α̃
    calculate b ← −C − α̃λ
    setup AMG for every few frames (e.g., 20).
    solve AΔλ = b using MGPCG solver
    Δx ← M^{-1}∇C^T Δλ
    λ ← λ + Δλ
    x ← x + ωΔx
    if timeBudgetExhausted or ||b|| < ε then
        break
    end if
end for
collision response
v ← (x − x_old)/Δt
```

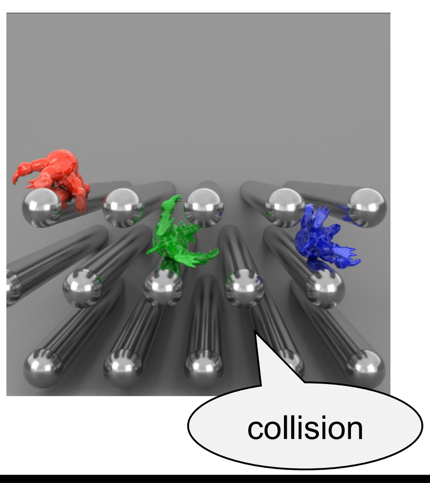
Results & Limitations

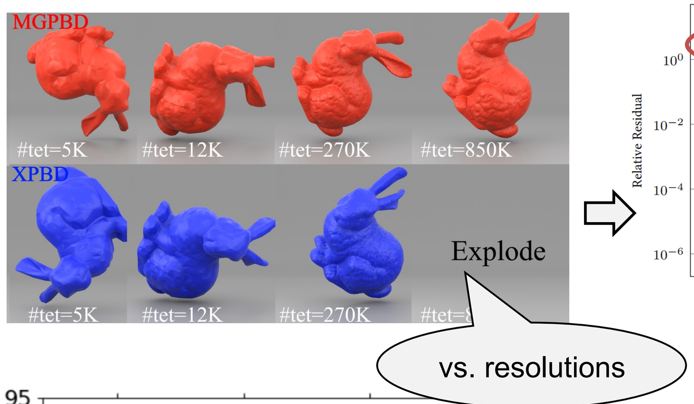


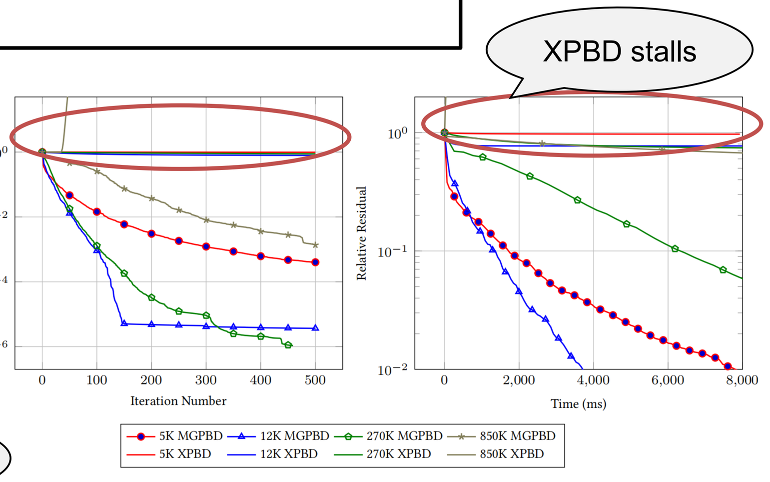
XPBD
- ✗ soft
- ✗ Crash

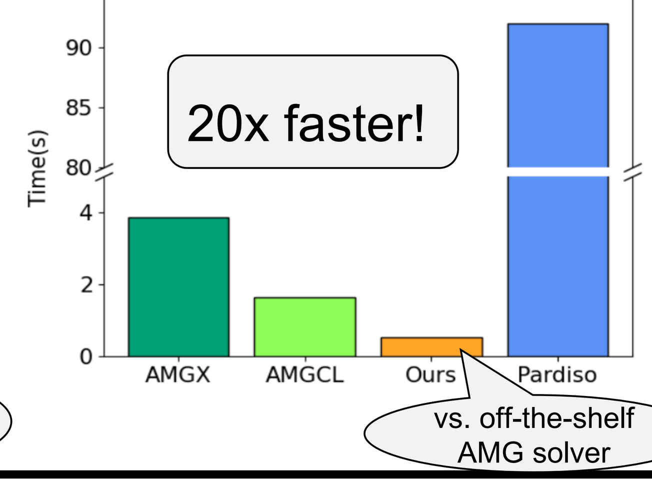
Ours
- ✓ Stiff
- ✓ Stable











Limitations

Global Solver

Every step is too **heavy**, whereas XPBD is known being lightweight and flexible.

AMG

Build P by A: lacks physical intuition.

References

1. Miles Macklin, Matthias Müller, and Nuttapong Chentanez. 2016. XPBD: position-based simulation of compliant constrained dynamics. *In Proceedings of the 9th International Conference on Motion in Games (Burlingame, California) (MIG '16).*
2. M. Macklin, K. Erleben, M. Müller, N. Chentanez, S. Jeschke, and T.Y.Kim. 2020. Primal/Dual Descent Methods for Dynamics. *ComputerGraphics Forum 39, 8 (2020), 89–100.*
3. Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. 2007. Efficient simulation of inextensible cloth. *ACM Trans. Graph. 26, 3 (July 2007), 49–es.*
4. Zangyueyang Xian, Xin Tong, and Tiantian Liu. 2019. A scalable galerkin multigrid method for real-time simulation of deformable objects. *ACM Trans. Graph. 38, 6, Article 162 (December 2019), 13 pages.*

taichi 泽森科工 ZENUSTECH