Breaking good : Fracture Modes for Realtime destruction

Silvia Sellán, Jack Luong, Leticia Mattos Da Silva, Aravind Ramakrishnan, Yuchuan Yang, and Alec Jacobson. 2022. ACM Trans. Graph. 1, 1, Article 1 (January 2022), 12 pages. https://doi.org/10.1145/3549540

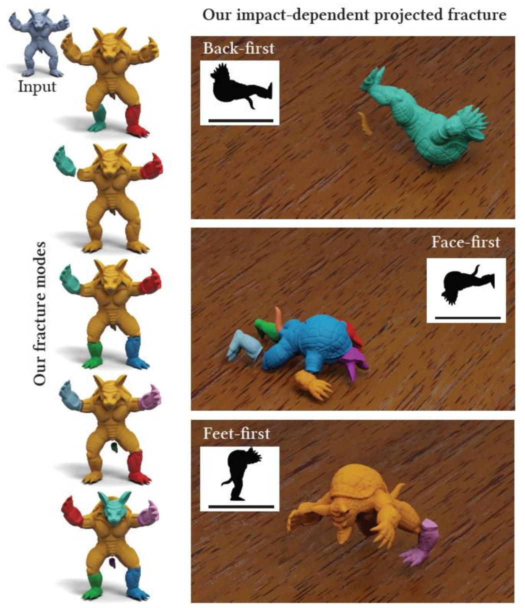
Seminar : Recent advances in 3D Computer Vision

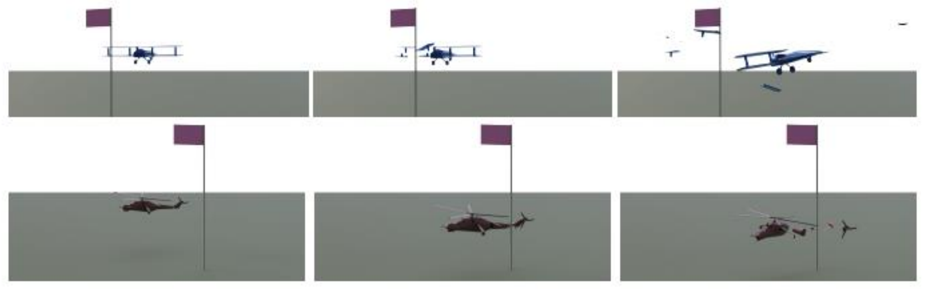
Nikhita Kurupakulu Venkat
Supervisor : Marvin Eisenberger

Picture : [SeLuSiRaYaJa]

Concept

- Introduce an object’s *fracture modes.*
- Obtain a pre fracture pattern – substitute existing methods for real-time applications with lower runtime costs and better realistic output.
- Obtain an impact dependent fracture pattern omitting the need for any kind of crack propagation simulation.
- Introduce and show advantages for these concepts.

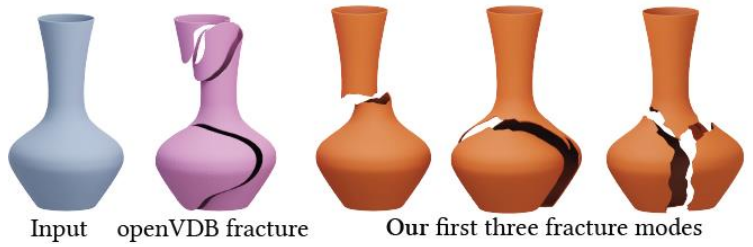




Picture : [SeLuSiRaYaJa]

Analogous existing work

- Modeling the fracture patterns of brittle fracture in a high performant way needs both high spatial and temporal resolution at the microsecond scale. *[Kirugulige et al. 2007]*
- Although convexity lacks realism, it can be an advantage for simulations.
- Voronoi methods generally miss obvious structural weaknesses and result in unconvincing fragments – Our fracture modes consistently output nonconvex fragments whose edges come from minimal stress displacement of the shape.



- Our method can add constraints to avoid fractures in certain areas
- Given a precomputed fracture pattern, one must decide which fractures are activated, which can be done using various strategies. Our Fracture Methods removes the need for heuristics or data based approaches.
- Existing methods to achieve realistic realistics are computationally expensive and are numerically fragile.

Picture : [SeLuSiRaYaJa]

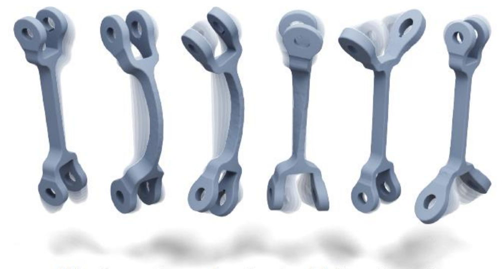
Introduction of the problem

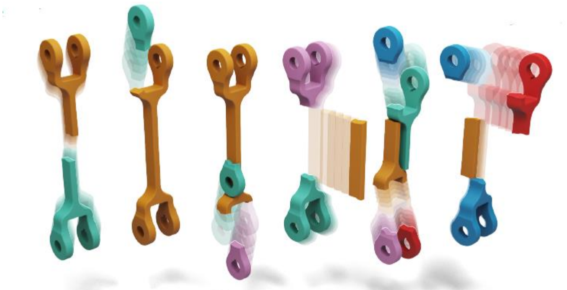
- Existing methods do not produce realistic results always.
- The methods that produce the most high-quality realistic results require hefty simulations which are too expensive for most realtime applications.
- Existing prefracture techniques use geometric heuristics that produce unrealistic patterns which do not consider the object's elastic profile or structural weaknesses.

Video : [SeLuSiRaYaJa]

Approach

- A method for prefracturing stiff brittle materials is presented that draws a direct analogy to a solid shape's elastic vibration modes.
- The shape's *fracture modes* are computed which algebraically span the shape's natural way of breaking.


Vibration modes = a shape's natural deformation patterns


Our **Fracture** modes = a shape's natural **breaking** patterns

- The unique and orthogonal modes of fracture are introduced in increasing order of a generalized notion of frequency.
- The first k fracture modes can be intersected against each other to define a prefracture pattern as a replacement to existing procedural methods.
- These impacts determined at runtime can be efficiently projected onto the linear space of precomputed fracture modes to obtain impact dependent fracture.

Picture : [SeLuSiRaYaJa]

Sparsified Eigen Problems

$$\underset{X^\top MX=I}{\operatorname{argmin}} \quad \frac{1}{2}\operatorname{trace}\left(X^\top LX\right) + \sum_{i=1}^{k} g(X_i)$$

Existing methods :

- Compressed nodes using a sparsity inducing l1 norm to compute localised solutions to Schrodingers equation.
- Usually ADDM methods do not apply to non convex problems but successful local convergence has been demonstrated.
- Iterative mode by mode optimization was used prominently.

Our proposed method :

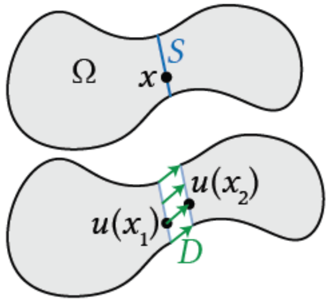
- Similar mode by mode fixed point iteration approach.
- Uniqueness : We do not consider the sparsity of the modal vector Xi itself but rather sparsity of the mode‘s continuity over the domain.

Fracture modes

Object‘s total strain energy $E_\Psi(u) = \int_\Omega \Psi(u, x)dx$

$\Omega \subset \mathbb{R}^{\mathbf{d}}$ and deformation map u : Ω -> $\mathbb{R}^{\mathbf{d}}$
$\boldsymbol{\Psi}$ is the strain energy density function evaluated at points x $\in \Omega$ in the undeformed object.

Fracture of deformation map u



- U is discontinous at S.
- Let x1 and x2 be undeformed positions on either side of the fracture fault.

Pointwise vector valued discontinuity $D(u, x \in S) = u(x_1) - u(x_2) \in \mathbb{R}^d$

Discontinuity energy associated with S $\int_{x \in S} \|D(u,x)\|^2 dx$

Total discontinuity energy associated with u $E_D(u) = \|D(u,S)\|_{2,1} := \sum_{i=1}^{p} \sqrt{\int_{S_i} \|D(u,x)\|^2 dx}$

Picture : [SeLuSiRaYaJa]

Fracture modes

Total energy $E(u) = E_\Psi(u) + \omega\, E_D(u)$

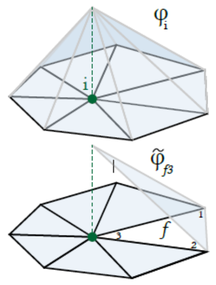
k lowest energy fracture modes can be evaluated as the arguments that would result in the least total energy

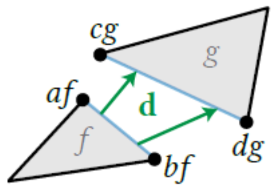
$$\{u^i\}_{i=1}^k = \underset{\{u^i\}_{i=1}^k}{\operatorname{argmin}} \sum_{i=1}^{k} E(u^i)\,, \qquad \text{s.t.} \quad \int_\Omega (u^i)^\top \rho u^j \, dx = \delta^{i,j}$$

The above equation can be discretized to obtain the fracture modes of a mesh (2D) as a traingle mesh Ω with n vertices and m faces. In this construction, the mesh‘s p interior edges correspond to the admissible fracture faults S_1…….S_p.

Fracture modes on meshes

- A triangular mesh Ω is considered with n vertices and m faces. The mesh's p interior edges correspond to admissible fracture faults $S_1 \ldots\ldots S_p$.
- Hat functions are by construction continuous. Usually, this is a good thing but we require functions with arbitrarily large co dimension one patches of discontinuities. Hence, we use an exploded mesh Ω~.





- Now, the object's total strain energy : $E_\Psi(\mathbf{u}) = \sum_{f=1}^{m} \int_f \Psi(u,x)\,dx$
- Contribution of edges on total discontinuity energy : $E_e(u) = \sqrt{\int_e \|D(u,x)\|^2 dx}$: $E_e(\mathbf{u}) = \sqrt{\frac{l}{2}\left(\left\|\mathbf{d}\left(\frac{+1}{\sqrt{3}}\right)\right\|_2^2 + \left\|\mathbf{d}\left(\frac{-1}{\sqrt{3}}\right)\right\|_2^2\right)}$
- k lowest energy fracture modes $\underset{\mathbf{U}^\top \tilde{\mathbf{M}}\mathbf{U}=\mathbf{I}}{\operatorname{argmin}} \quad \frac{1}{2}\operatorname{trace}\left(\mathbf{U}^\top \mathbf{Q}\mathbf{U}\right) + \omega \sum_{i=1}^{k} \left(E_D(\mathbf{U}_i)\right)$

Picture : [SeLuSiRaYaJa]

Optimization

Existing method

- ICCM computes the modes sequentially
- Assuming the first $i-1$ columns of U have been computed. In the original ICCM formulation, the process for finding the ith column, U_i proceeds by choosing a random unit-norm vector c, then repeatedly solving

$$\begin{aligned} \mathbf{U}_i \leftarrow & \operatorname*{argmin}_{\mathbf{u}} \frac{1}{2}\mathbf{u}^\top \mathbf{Q}\mathbf{u} + \omega E_D(\mathbf{u}) \\ & \text{subject to} \begin{bmatrix} \mathbf{U}_1^\top \\ \vdots \\ \mathbf{U}_{i-1}^\top \\ \mathbf{c}^\top \end{bmatrix} \tilde{\mathbf{M}}\mathbf{u} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \end{aligned}$$

and updating

$$\mathbf{c} \leftarrow \frac{\mathbf{U}_i}{\sqrt{\mathbf{U}_i^\top \tilde{\mathbf{M}} \mathbf{U}_i}}$$

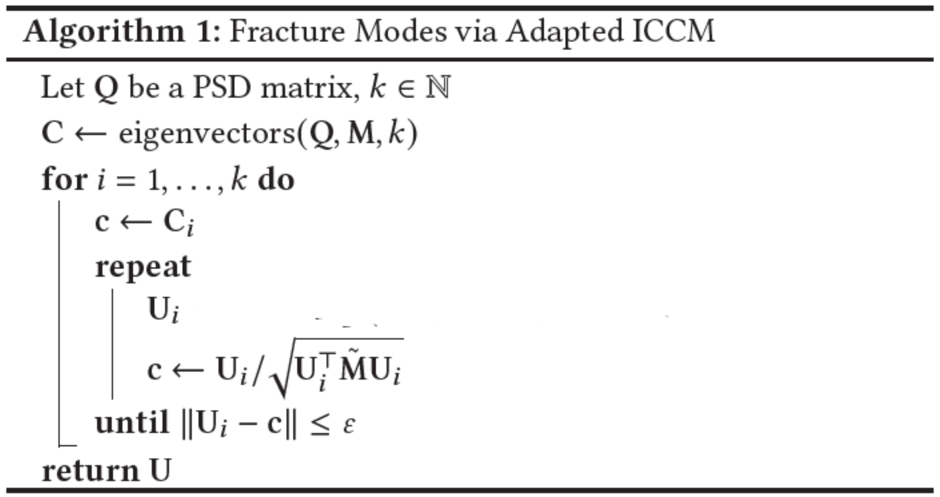
Disadvantages :

- Random initialization of c results in non determinism.
- It also sometimes results in sub optimal local minima and a large number of inner iterations.

Optimization

Our method:

- c is initialized to the ith continous eigenvectors of Q.
- The k initial vectors are computed using the ‘SCIPY‘ wrapper for the sparse eigensolver ARPACK.

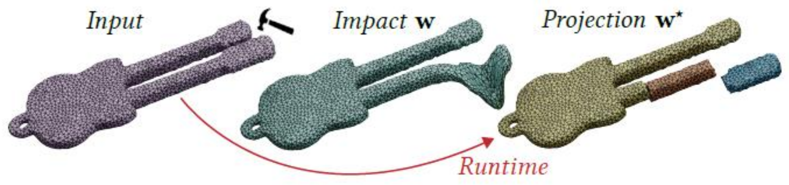


Algorithm 1: Fracture Modes via Adapted ICCM

```
Let $\mathbf{Q}$ be a PSD matrix, $k \in \mathbb{N}$
$\mathbf{C} \leftarrow$ eigenvectors$(\mathbf{Q}, \mathbf{M}, k)$
for $i = 1, \dots, k$ do
    $\mathbf{c} \leftarrow \mathbf{C}_i$
    repeat
        $\mathbf{U}_i$
        $\mathbf{c} \leftarrow \mathbf{U}_i / \sqrt{\mathbf{U}_i^\top \tilde{\mathbf{M}} \mathbf{U}_i}$
    until $\|\mathbf{U}_i - \mathbf{c}\| \leq \varepsilon$
return $\mathbf{U}$
```

Impact dependent fracture

- The columns of U form an orthonormal basis of the lowest-energy k-dimensional subspace of possible fractures of Ω. Which implies that we can precompute an object's prefecture pattern directly after its design.
- Any detected impact can be projected onto our modes to obtain an impact dependent realtime fracture.



- If a collision is detected between Ω and another object, with contact point p and normal n, the exploded vertex wise impact vector w $\mathbf{w}_{cf} = g(p, v_{cf})\,\vec{n}, \quad \forall c = 1, 2, 3,\ f = 1, \ldots, m$, g is a filter that vanishes when vcf is far from p.
- Projected Impact : $\mathbf{w}^{\star} = \sum_{i=1}^{k} \mathbf{U}_i \mathbf{U}_i^{\top} \tilde{\mathbf{M}} \mathbf{w} = \sum_{i=1}^{k} \mathbf{U}_i \mathbf{U}_i^{\top} \tilde{\mathbf{M}} \mathbf{g} \vec{n}$ such that $\mathbf{g} = \left(\tilde{\mathbf{M}} - \tau \tilde{\mathbf{L}}\right)^{-1} \tilde{\mathbf{M}} \delta_p$ But, computing g would require

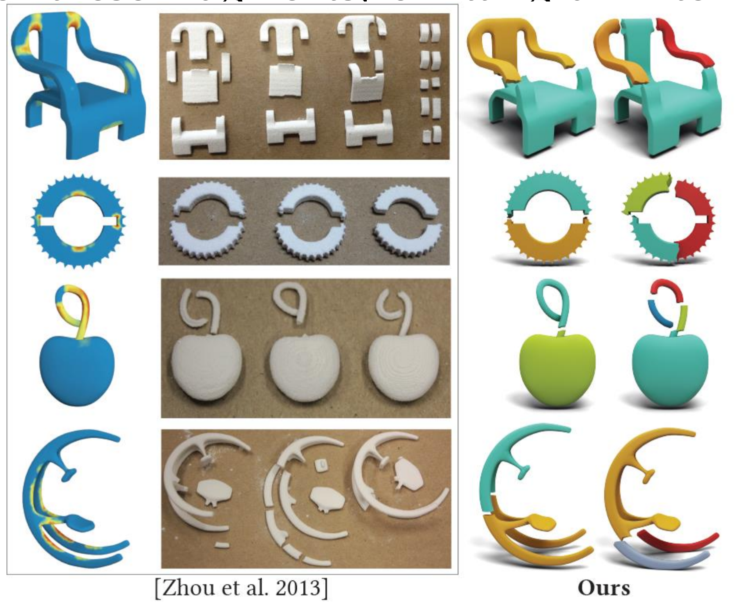
solving a linear system at runtime. This can be avoided by precomputing $\mathbf{A}_i = \mathbf{U}_i^{\top} \tilde{\mathbf{M}} \left(\tilde{\mathbf{M}} - \tau \tilde{\mathbf{L}}\right)^{-1} \tilde{\mathbf{M}}$

- Now, only a matrix multiplication is required at runtime $\mathbf{w}^{\star} = \sum_{i=1}^{k} \mathbf{U}_i \mathbf{A}_i \delta_p \vec{n}$
- The choice of g makes w* linearly dependent on the impact.

Picture : [SeLuSiRaYaJa]

Efficient implementation for realtime fracture

- In 3D, our computation requires a tetrahedral mesh but realtime applications prefer triangle meshes for input and output. But, the sparsity inducing discontinuity norm results in fracture modes which are continuous across most pairs of neighboring tetrahedra.
- It is not necessary to keep the whole tetrahedral mesh and hence we can determine the connected components. The boundary of each component is a solid triangle mesh of a fracture fragment.
- We can pre restrict the projection to the vertices on the boundary of these fragments, omitting all internal vertices and the tetrahedral connectivity.



Picture : [SeLuSiRaYaJa]

Simpler nested cages

Many problems that arise at the precomputation stage can be resolved by working with a tetrahedral coarse cage nesting the input.

Existing methods:

- There exist methods(NESTED CAGES) to produce tight fitting cages, but they take long runtimes, cause potential failure and may result in a surface mesh that might cause consecutive tetrahedralization.

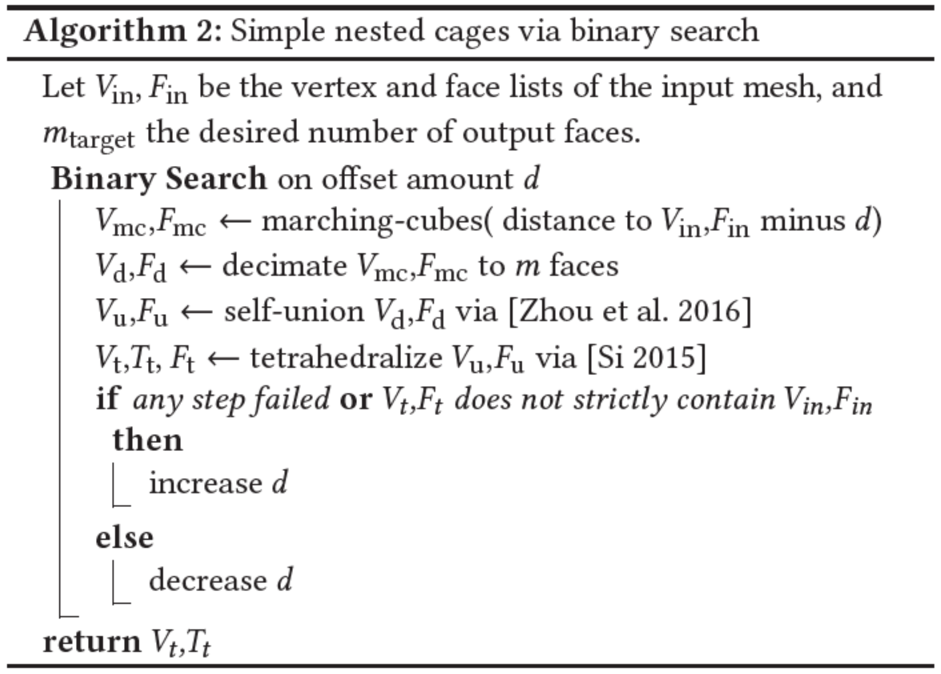
Our proposed method:

- We introduce a simpler caging method inspired by a method of Ben-Chen et al. [2009].
- Similar to NESTED CAGES, the output cage will strictly contain the input but also, we ensure that this cage can successfully be tetrahedralized and not just in theory.
- This method provides a different point on the Pareto frontier of tightness-vs-utility.

Picture : [SeLuSiRaYaJa]

Simpler nested cages

- Fracture modes and solid fragment components on the cage's tetrahedralization can be transferred to the true input geometry by intersecting each component against the input mesh.
- Hence, the exterior surface of each fragment component is exactly a subset of the input mesh.

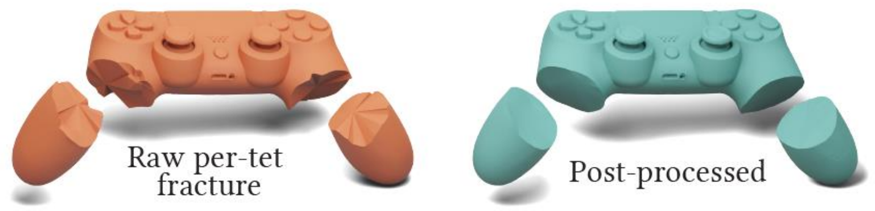


Algorithm 2: Simple nested cages via binary search

```
Let $V_{\text{in}}$, $F_{\text{in}}$ be the vertex and face lists of the input mesh, and
$m_{\text{target}}$ the desired number of output faces.
Binary Search on offset amount $d$
    $V_{\text{mc}}$,$F_{\text{mc}}$ $\leftarrow$ marching-cubes( distance to $V_{\text{in}}$,$F_{\text{in}}$ minus $d$)
    $V_{\text{d}}$,$F_{\text{d}}$ $\leftarrow$ decimate $V_{\text{mc}}$,$F_{\text{mc}}$ to $m$ faces
    $V_{\text{u}}$,$F_{\text{u}}$ $\leftarrow$ self-union $V_{\text{d}}$,$F_{\text{d}}$ via [Zhou et al. 2016]
    $V_{\text{t}}$,$T_{\text{t}}$, $F_{\text{t}}$ $\leftarrow$ tetrahedralize $V_{\text{u}}$,$F_{\text{u}}$ via [Si 2015]
    if any step failed or $V_t$,$F_t$ does not strictly contain $V_{in}$,$F_{in}$
     then
        increase $d$
    else
        decrease $d$
return $V_t$,$T_t$
```

Smoothing internal surfaces

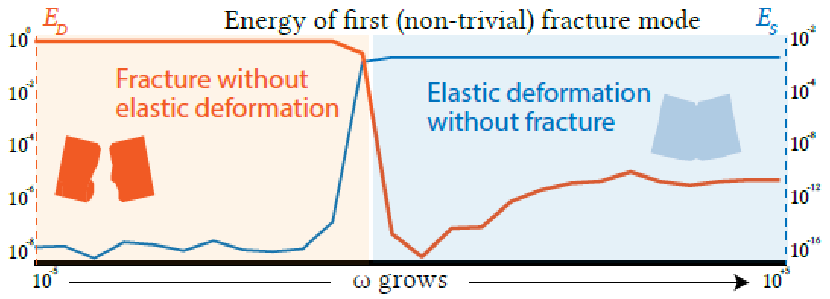
- By our construction, the fracture boundaries will follow faces of the tetrahedral mesh used for their computation. This reveals that the frequency is proportional to the mesh resolution.
- This can be alleviated by treating each extracted per tet component membership as a one-hot vector field, which can be averaged onto mesh vertices which is stored as a matrix Z. $Z \leftarrow (M + \lambda L)^{-1}(MZ)$
- By the nature of the Laplacian, the equation will push our faults towards smooth surfaces.
- Naturally, crystalline materials break along smooth surfaces but materials like wood or clay do not necessarily similar to those produced by our method. This is a limitation shared with all mesh-based fracture algorithms.

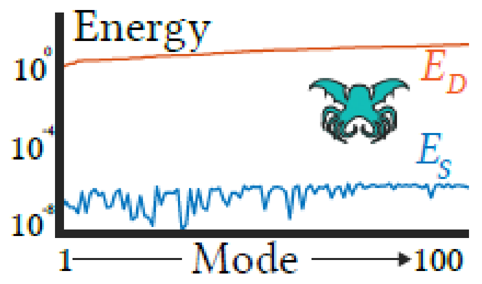


Picture : [SeLuSiRaYaJa]

Choice of strain energy

- The only requirement on strain energy density Ψ is that we can evaluate its second order approximation near the rest configuration represented by the positive semi definite Hessian Q.
- When the discontinuity energy is zero, then we have recovered the usual linear elastic vibration modes.
- When the strain energy is zero, then we start to see sparse fractures and also see that each fracture fragment undergoes its own zero strain energy transformation.





- Numerically, this implies that the precise choice of Q is irrelevant and only its null space matters although, physically, it aligns with our fracture modes with the traditional definition of stiff brittle fracture.

Picture : [SeLuSiRaYaJa]

Choice of strain energy

- So, we work with a strain energy that only admits translational motions in its null space, namely . The Hessian is simp $\Psi(u, x) = \|\nabla u(x)\|^2$ lacian matrix L~ repeated for each spatial coordinate. $Q = I_d \otimes \tilde{L}$

Efficient precomputation

- The strain energy being zero in all our modes means all vertices belonging to a single element undergo identical deformation.
- This observation can be transformed into an assumption, and we may store deformations solely as elements, reducing our number of variables by a factor of d+1.
- Additionally, allowing only per element deformation also makes our vector discontinuity D necessarily constant along element boundaries.

Results

- Machine used to record timings – 2020 13-inch MacBook Pro with 16GB memory and 2.3GHz Quad-Core Intel Core i7 processor.
- To produce animations, a traditional HOUDINI(SideFX 2020) fracture simulation workflow was used exchanging the usual Voronoi or openVDB fracture nodes for our own fractured meshes.
- Our algorithm's only parameters are the tolerances ε and σ, which we fix at $\varepsilon = 10^{-10}$ and $\sigma = 10^{-3}$

The proposed algorithm works in two steps :

Step 1 : Precomputing a given shape's fracture modes.

- This step takes place offline, following algorithm 1.
- Each mode takes between 0.5 to 12 seconds to compute in our meshes, which have between 3,000 and 15,000 tetrahedra.

Results

Step 2 : Impact projection step

- Only part that happens at runtime.
- Complexity : O($k\tilde{n}$)
- Complexity of other elements of the projection step : O(p).
- In our examples, $\tilde{n}$ is between 1,000 and 10,000 and we compute between k = 20 and k = 40 fracture modes, meaning our full runtime step requires between 0.1 and 1 million floating point operations.
- Implementation took between one and two milliseconds to carry out this step

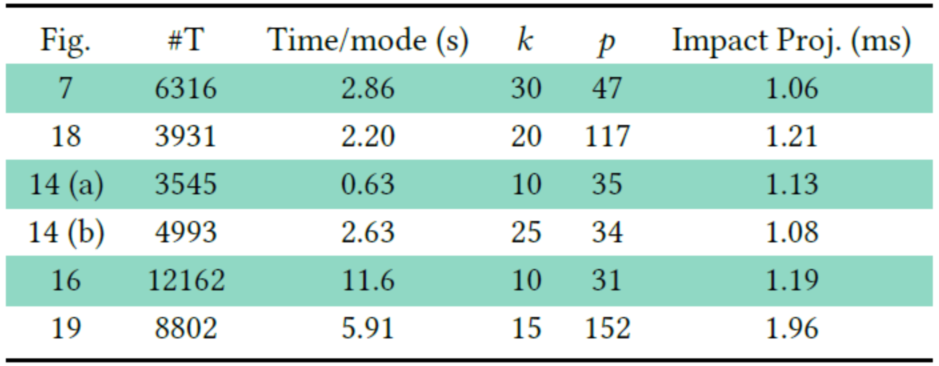
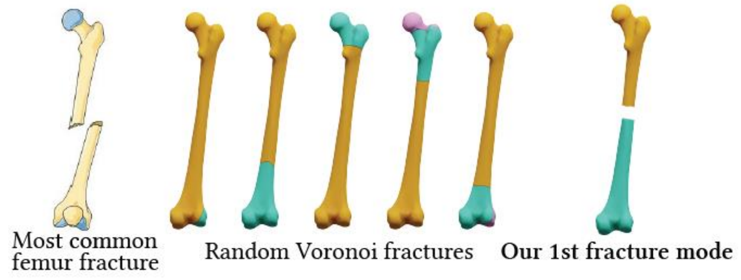
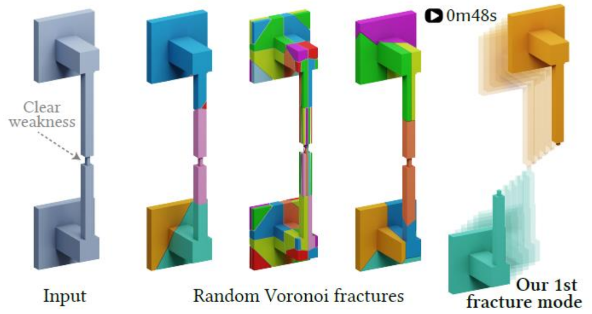


Fig.	#T	Time/mode (s)	k	p	Impact Proj. (ms)
7	6316	2.86	30	47	1.06
18	3931	2.20	20	117	1.21
14 (a)	3545	0.63	10	35	1.13
14 (b)	4993	2.63	25	34	1.08
16	12162	11.6	10	31	1.19
19	8802	5.91	15	152	1.96

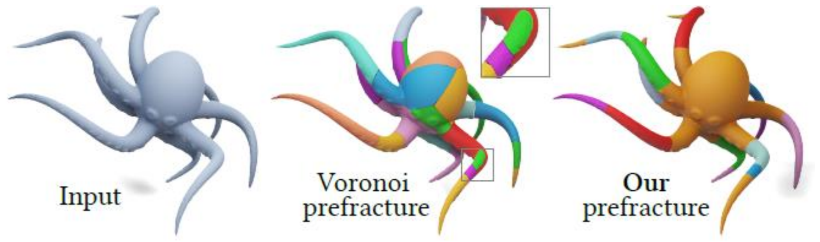
Summary

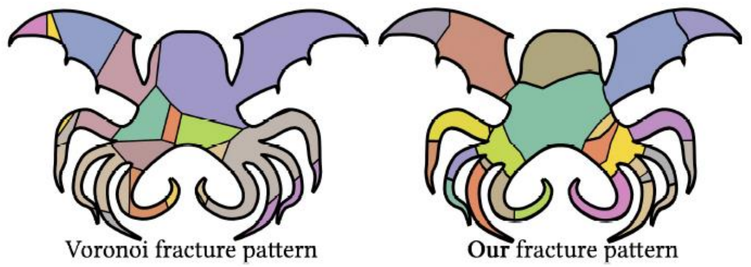
- Our proposed fracture modes naturally identify the regions of a shape that are geometrically weak.





- Voronoi based prefecture methods result in convex, unrealistic and easily recognizable pieces, while our fracture modes are realistic and can produce a much wider set of shapes.



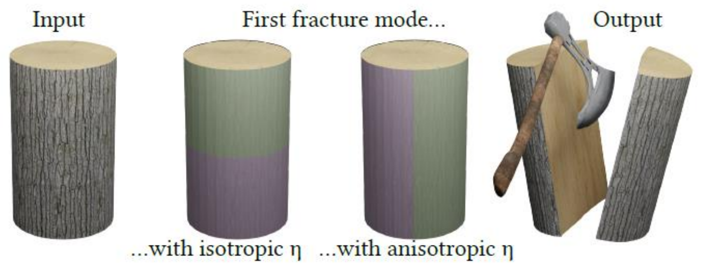


Picture : [SeLuSiRaYaJa]

Summary

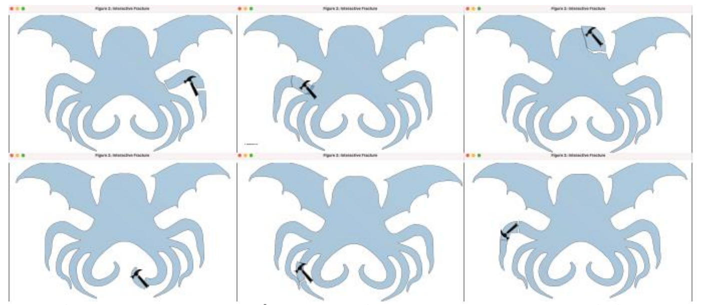
- Heterogenous and anisotropic materials are modelled using a vector field $\eta: \Omega \rightarrow R^d$ to the discontinuity energy equation.

$$E_D(u) = \sum_{i=1}^{p} \sqrt{\int_{S_i} \|\eta(x) \circ D(u,x)\|^2 dx}$$



Fracture simulations

- Our proposed method is ideal for use in interactive applications. The user can select the impact position to obtain different breaking patterns.

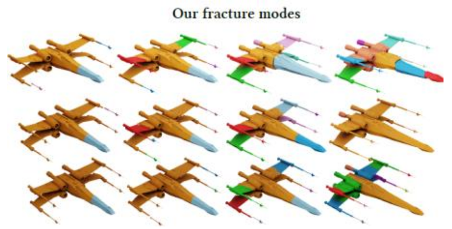


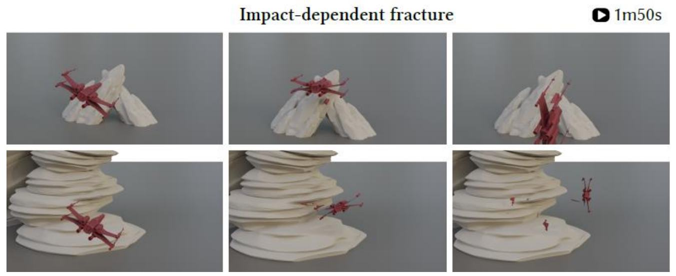
Picture : [SeLuSiRaYaJa]

Summary

Fracture simulations

- An excellent use of these fracture modes are video games. A player can see different fracture behaviors depending on the received impact.



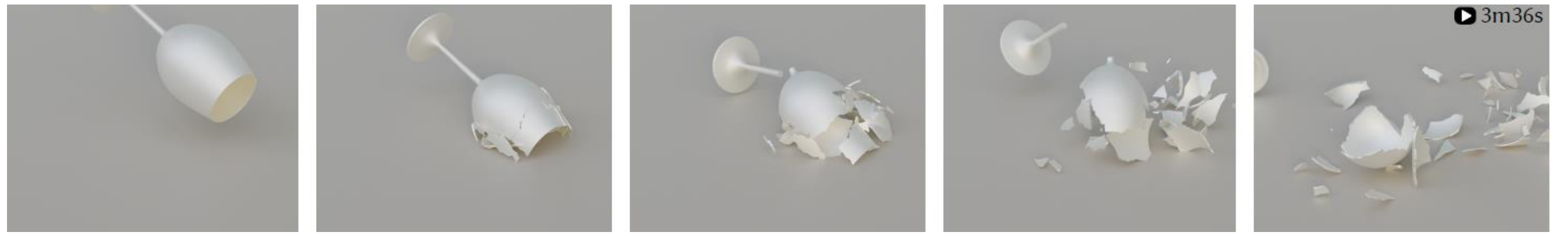


- These fracture modes can be used for any realtime fracture application.

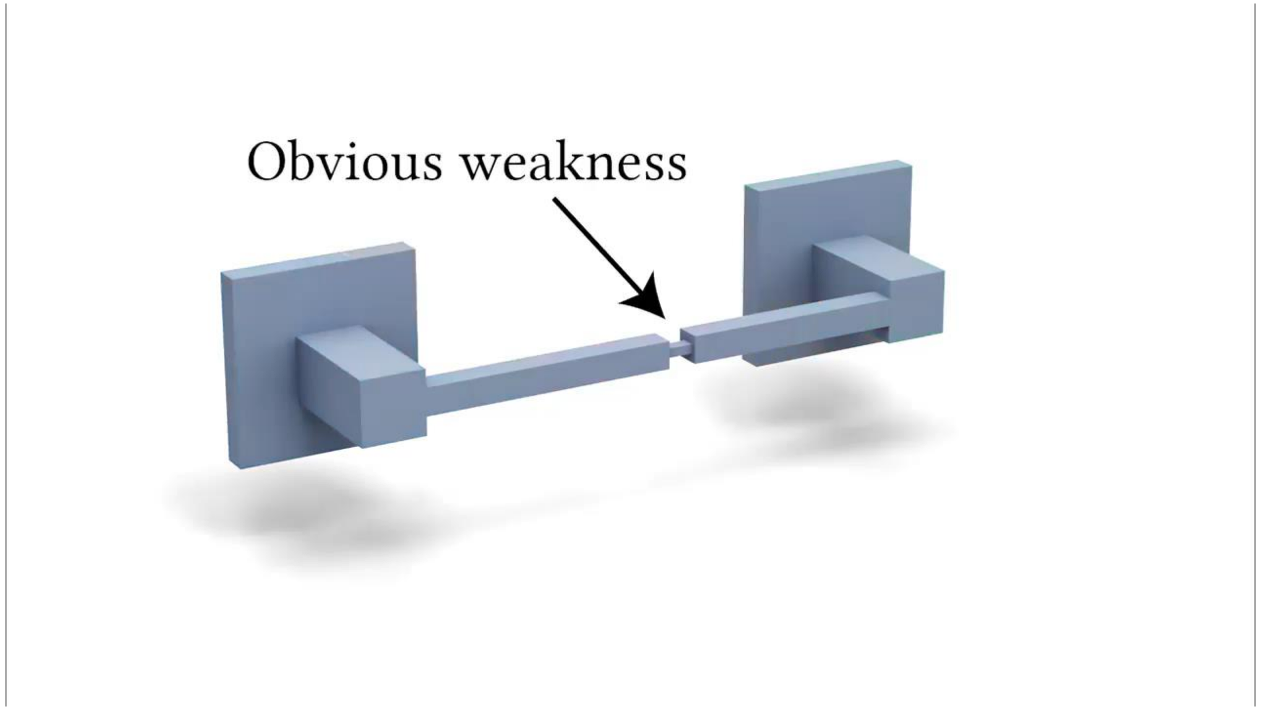
Input
Our fracture modes
Our impact-dependent projected fracture
Back-first
Face-first
Feet-first
3m15s

Picture : [SeLuSiRaYaJa]

Summary



- A glass cup shattering results in many nonconvex pieces which would be impossible to obtain with Voronoi decomposition.



Picture and Video : [SeLuSiRaYaJa]

Limitations

- On experimenting with MANOPT, it was observed that the performance was significantly lower than the proposed method. For very large meshes in realtime applications, the projection step could exceed CPU usage allowances for realtime applications.
- The fracture modes are global in nature – they create relations between regions of the object that will not typically fracture together.
- Our algorithm is designed to fit into realtime rigid body simulations like those seen in video games. Hence, the outputs do not contain partial fractures.
- Secondary fractures were not included in the simulations as computing a new set of fracture modes for each piece would exceed realtime constraints.

2nd fractures

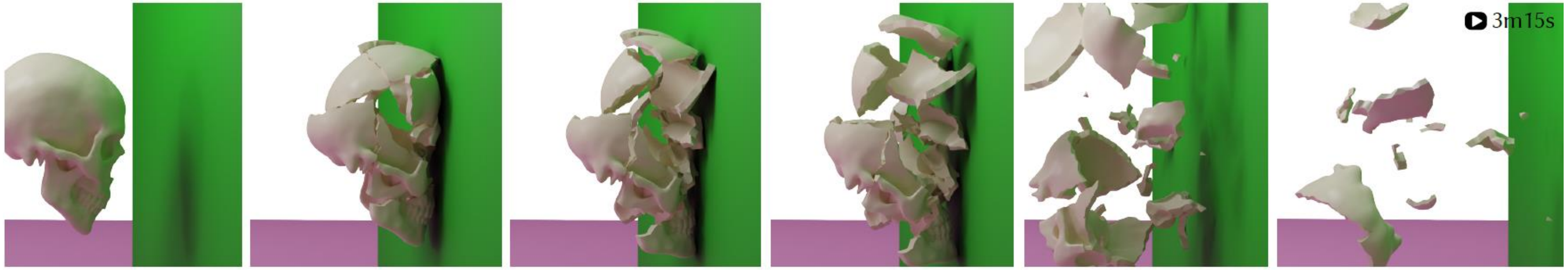
Picture : [SeLuSiRaYaJa]

Scope for future work

- Physical material properties can be incorporated in the mode computation, for instance, by treating tangential and normal discontinuities differently.

Thank you!

Any Questions?



Picture : [SeLuSiRaYaJa]