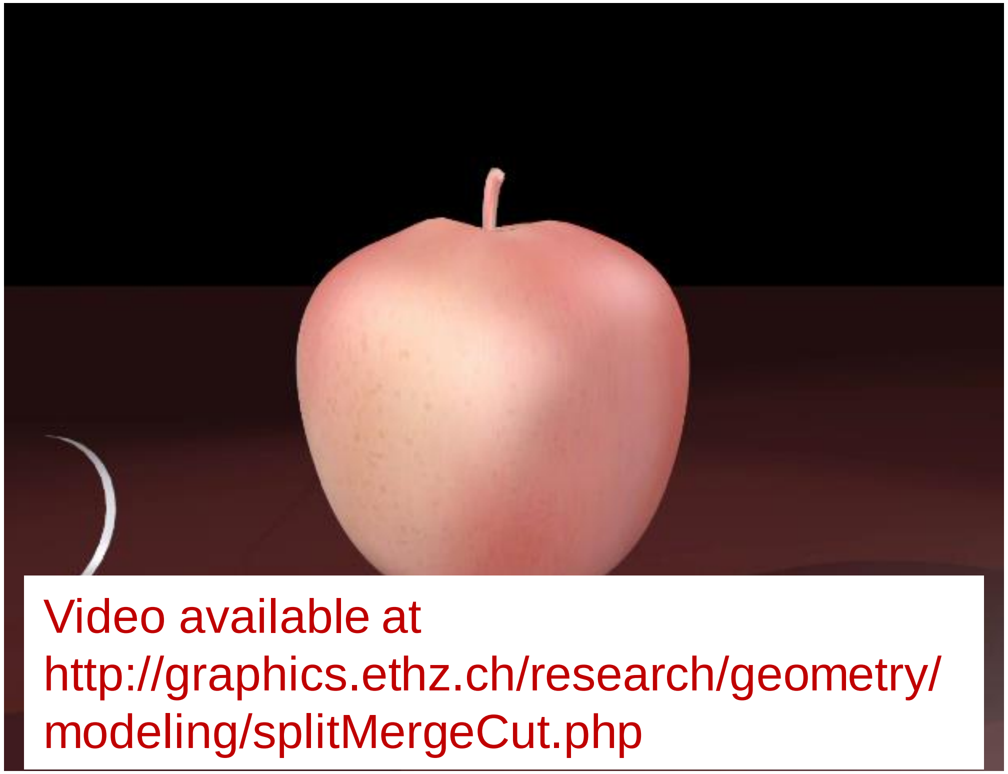
Physically-based Simulation of Cuts in Deformable Bodies

Jun Wu, Rüdiger Westermann, Christian Dick

Computer Graphics & Visualization Group
TU München, Germany

Virtual Cutting in Computer Animation

- Applications: computer games, visual effects

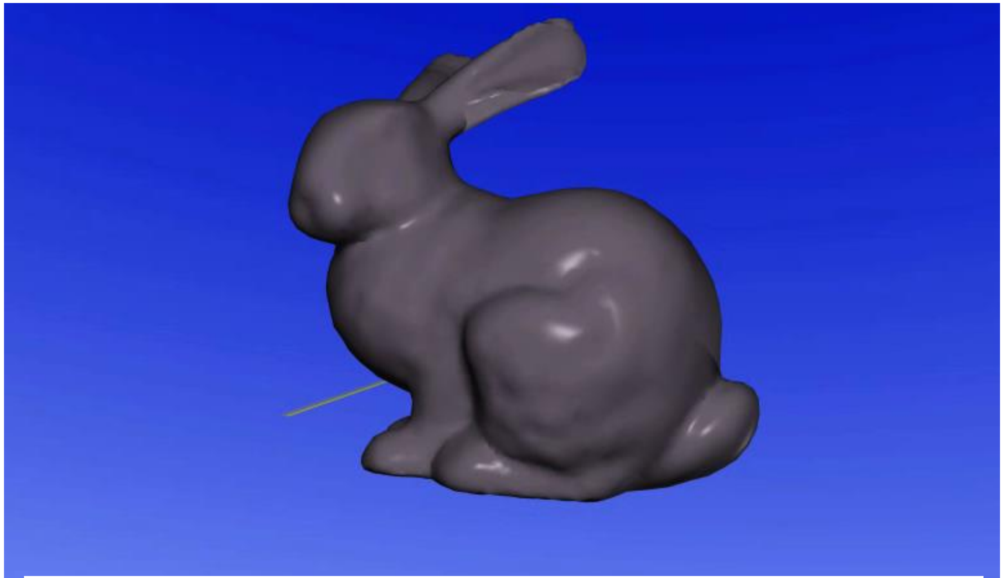


Meshfree method

[Steinemann et al. 2006]

Virtual Cutting in Computer Animation

- Applications: computer games, visual effects



Video available at
http://graphics.ethz.ch/publications/papers/papers.php

Polyhedral finite element method

[Wicke et al. 2007]

Virtual Cutting in Surgery Simulation

- Applications: surgery skill training, pre-operative planning

Video available at
http://hal.inria.fr/hal-00686056

Surgery simulation on a patient data set

[Courtecuisse et al. 2010]

SHACRA team at INRIA

Motivation of the Report

❖ Provide an overview of recent virtual cutting techniques

Reference	Geometry	Deformation	Solver	Scenario	Remark
Bielser et al. [BMG99, BG00, BGTG04]	Tet., refinement	Mass-spring	Explicit/Semi-implicit	Interactive	Basic tet. refinement
Cotin et al. [CDA00]	Tet., deletion	Tensor-mass	Explicit	Interactive	Hybrid elastic model
Mor & Kanade [MK00]	Tet., refinement	FEM	Explicit	Interactive	Progressive cutting
Nienhuys et al. [NFvdS00, NFvdS01]	Tet., boundary splitting/snapping	FEM	Static (CG solver)	Interactive	FEM with a CG solver
Bruyns et al. [BSM*02]	Tet., refinement	Mass-spring	Explicit	Interactive	An early survey
Steinemann et al. [SHGS06]	Tet., refinement + snapping	Mass-spring	Explicit	Interactive (Fig. 13 a)	Hybrid cutting
Chentanez et al. [CAR*09]	Tet., refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 d)	Needle insertion
Courtecuisse et al. [CJA*10, CAK*14]	Tet., deletion/refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 c,e)	Surgery applications
Molino et al. [MBF04]	Tet., duplication	FEM	Mixed explicit/implicit	Offline	Basic virtual node algorithm
Sifakis et al. [SDF07]	Tet., duplication	FEM		Offline (Fig. 12 a)	Arbitrary cutting
Jeřábková & Kuhlen [JK09]	Tet.	XFEM	Implicit (CG solver)	Interactive	Introduction of XFEM
Turkiyyah et al. [TKAN09]	Tri.	2D-XFEM	Static (direct solver)	Interactive	XFEM with a direct solver
Kaufmann et al. [KMB*09]	Tri./Quad.	2D-XFEM	Semi-implicit	Offline (Fig. 12 c)	Enrichment textures
Frisken-Gibson [FG99]	Hex., deletion	ChainMail	Local relaxation	Interactive	Linked volume
Jeřábková et al. [JBB*10]	Hex., deletion	CFEM		Interactive	CFEM
Dick et al. [DGW11a]	Hex., refinement	FEM	Implicit (multigrid)	Offline/Interactive (Fig. 12 d)	Linked octree, multigrid solver
Seiler et al. [SSSH11]	Hex., refinement	FEM	Implicit	Interactive	Octree, surface embedding
Wu et al. [WDW11, WBWD12, WDW13]	Hex., refinement	CFEM	Implicit (multigrid)	Interactive (Fig. 13 b, f)	Collision detection for CFEM
Wicke et al. [WBG07]	Poly., splitting	PFEM	Implicit	Offline (Fig. 12 b)	Basic polyhedral FEM
Martin et al. [MKB*08]	Poly., splitting	PFEM	Semi-implicit	Offline	Harmonic basis functions
Pauly et al. [PKA*05]	Particles, transparency	Meshfree	Explicit	Offline	Fracture animation
Steinemann et al. [SOG06]	Particles, diffraction	Meshfree		Offline/Interactive (Fig. 12 e)	Splitting fronts propagation
Pietroni et al. [PGCS09]	Particles, visibility	Meshfree		Interactive	Splitting cubes algorithm

Motivation of the Report

❖ Provide an overview of recent virtual cutting techniques
- Share our experience and understanding on this topic

Video available at http://wwwcg.in.tum.de/research/research/publications/2011/a-hexahedral-multigrid-approach-for-simulating-cuts-in-deformable-objects.html

Hexahedral finite element method on an octree grid

Armadillo:
500k elements,
10 seconds per frame

[Dick et al. 2011]

Motivation of the Report

- ❖ Provide an overview of recent virtual cutting techniques
- Share our experience and understanding on this topic

Video available at http://wwwcg.in.tum.de/research/research/projects/real-time-haptic-cutting.html

Haptic cutting of
high-resolution soft tissues

Liver:
15 fps
3k DOFs (170k elements)

[Wu et al. 2014]

Motivation of the Report

- Provide an overview of recent virtual cutting techniques
- Share our experience and understanding on this topic
- Discuss and identify future research problems
  - How to realistically simulate various cutting effects?

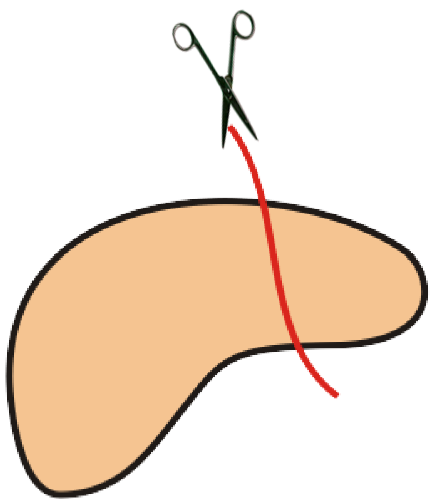
Cutting in hospitals

Cutting in kitchens

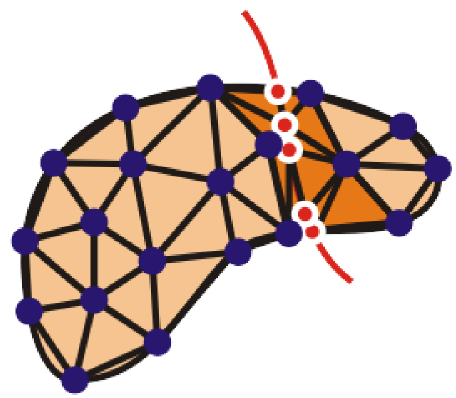
Images removed due to copyright

Virtual Cutting from a Computational Point of View

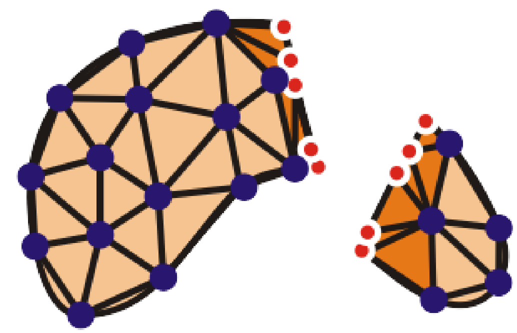
- ❖ Incorporation of cuts into the computational model
- ❖ Deformable body simulation



2D illustration of cutting process



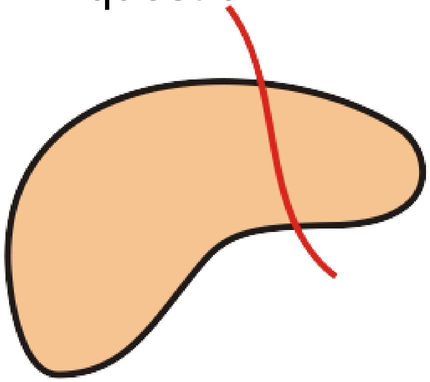
Mesh-based modeling of cuts



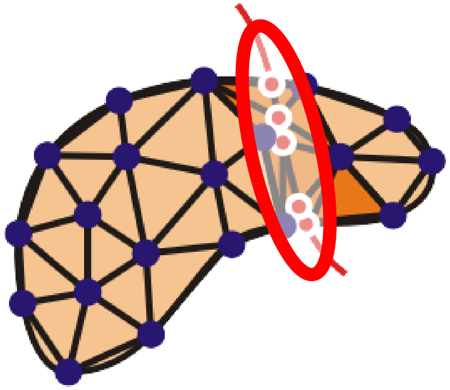
FE simulation of deformation

Virtual Cutting from a Computational Point of View

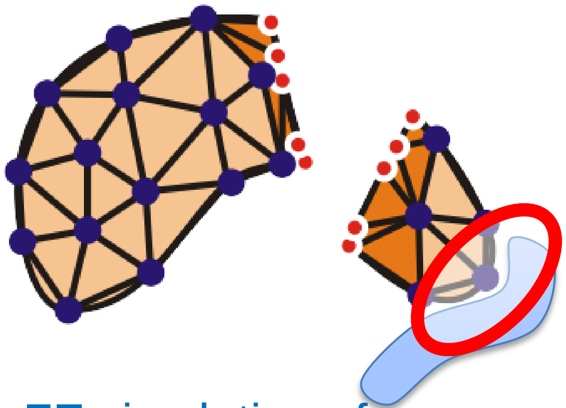
- ❖ Incorporation of cuts into the computational model
- ❖ Deformable body simulation
- o Detection and handling of collisions
  - o Collision detection: STAR by Teschner et al. 2005
  - o Realistic contact handling between a scalpel and a soft object: Open question



2D illustration of cutting process



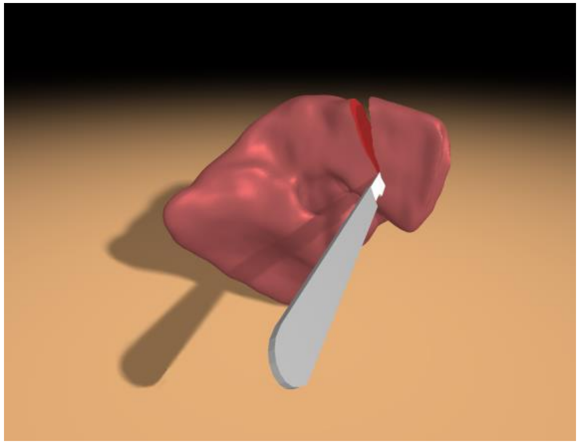
Mesh-based modeling of cuts



FE simulation of deformation

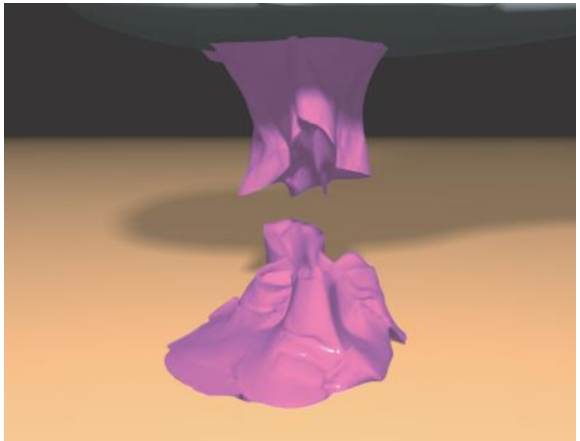
Cutting & Fracturing

- Cutting
  - Controlled separation of a physical object
  - As a result of an acutely directed force, exerted through sharp tools
- Fracturing
  - Cracking / breakage of (hard) objects
  - Under the action of stress



[Wu et al. 2014]

Cutting example

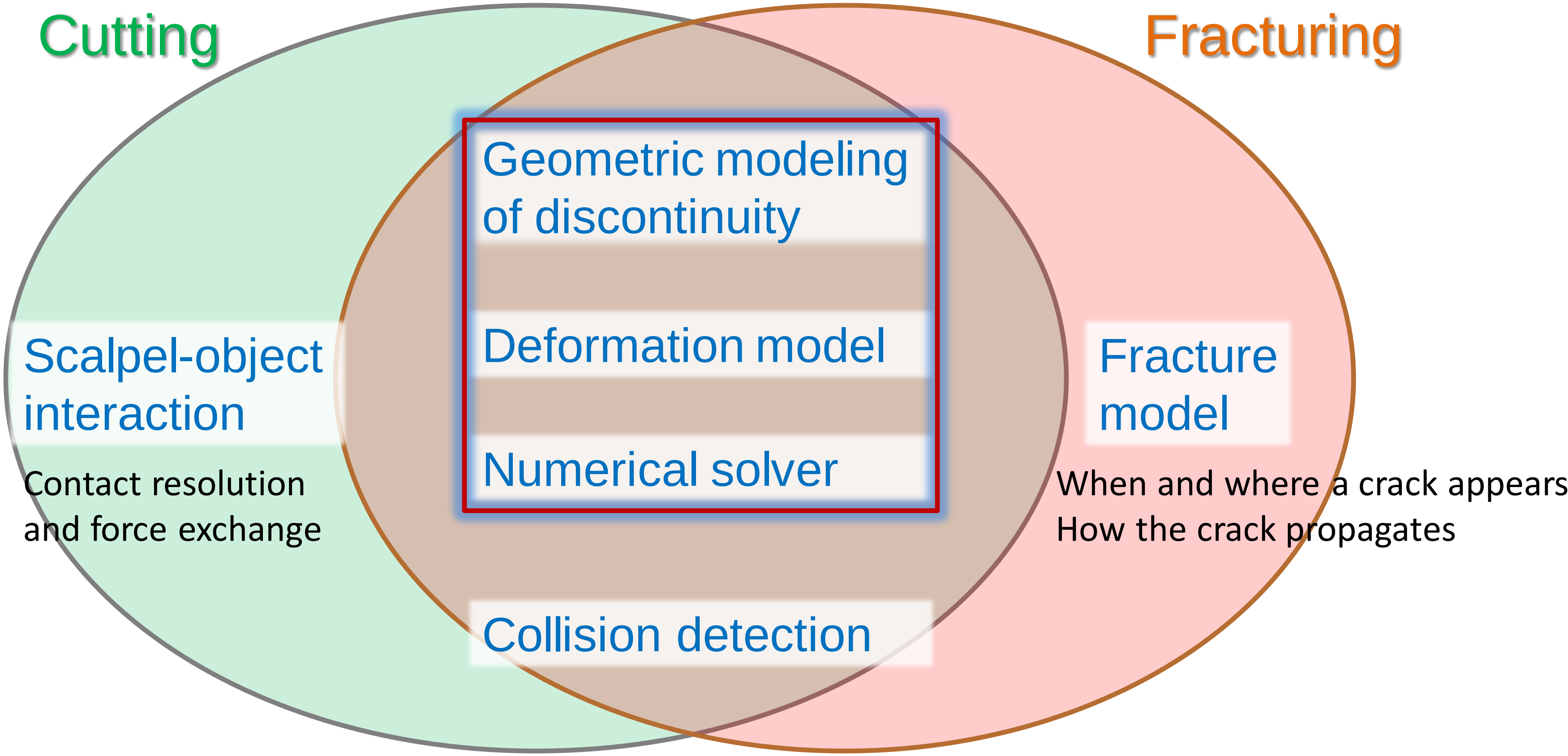


[Pauly et al. 2005]

Fracturing example

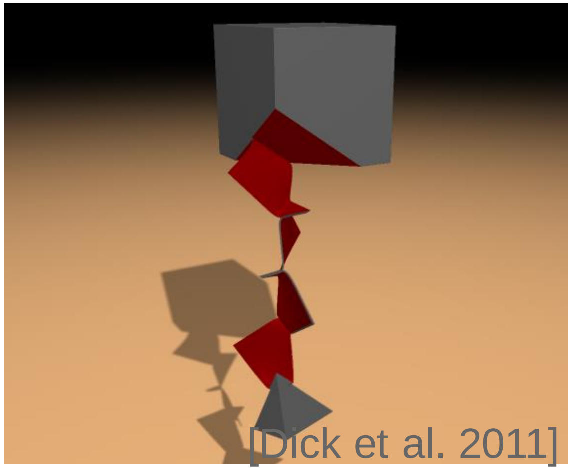
Cutting & Fracturing

- from a computational point of view



Challenges

- Physical accuracy
  - Ability to represent arbitrarily-shaped cuts in geometry and topology
  - Ability to predicate the dynamic behavior
- Solutions:
  - Dynamic local refinement of different spatial discretizations
  - Various finite element methods



[Dick et al. 2011]

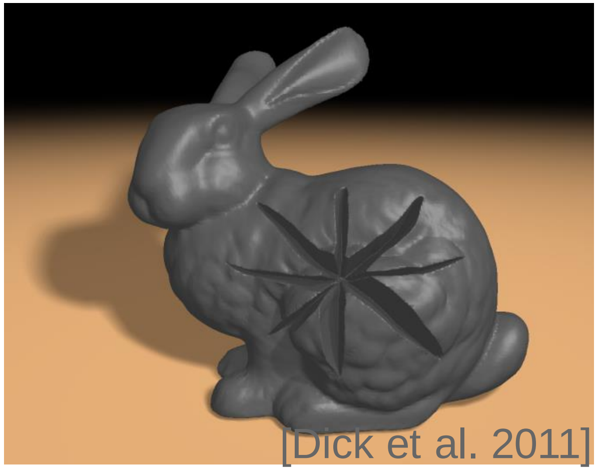
[Steinemann et al. 2006]

Examples of complicated cuts

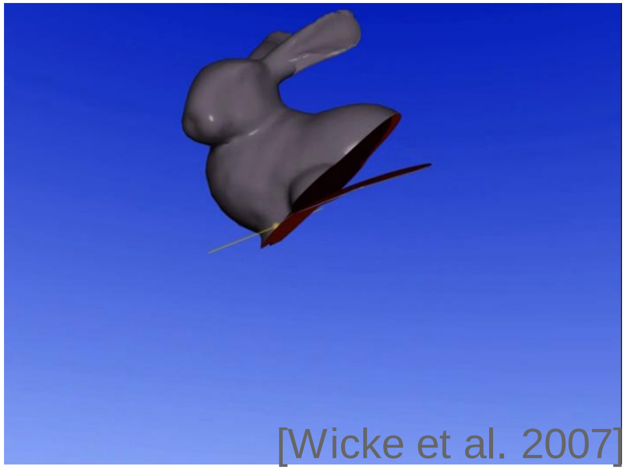
Challenges

- Physical accuracy
- Robustness
  - Numerical stability in complicated scenarios, e.g., repeated cutting, thin slicing

- Solution: to avoid ill-shaped elements, e.g., by virtual node algorithm, hexahedral discretization



Repeated cutting

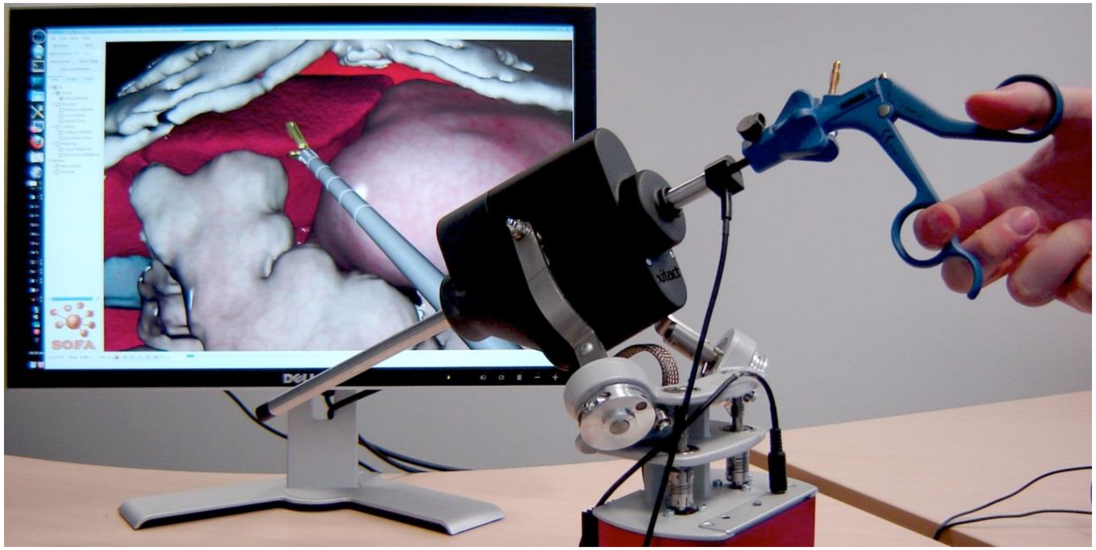


Thin slicing

Challenges

- Physical accuracy
- Robustness
- Computation efficiency

- Solutions: reducing #DOFs, efficient solvers, parallelization



Surgery simulation
with haptic feedback

[Courtecuisse et al. 2010]

Outline

follows the structure of the report

- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation of Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

❖ Principles and differences, not the implementation details
❖ 2D illustrations, but applicable to 3D volumetric cutting

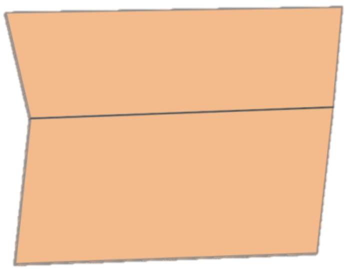
Outline

follows the structure of the report

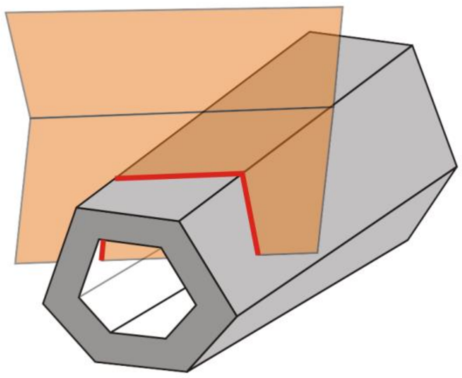
- Introduction
- Mesh-based Modeling of Cuts
  - Modeling of the Cutting Process
  - Tetrahedral Meshes
  - Hexahedral Meshes
  - Polyhedral Meshes
  - Discussion on Discretizations
- Finite Element Simulation for Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Modeling of the Cutting Process

- Detect intersections between the volumetric mesh **(**the deformable object**)** and a surface mesh (cutting surface)
  - Edge-face test



Cutting surface mesh

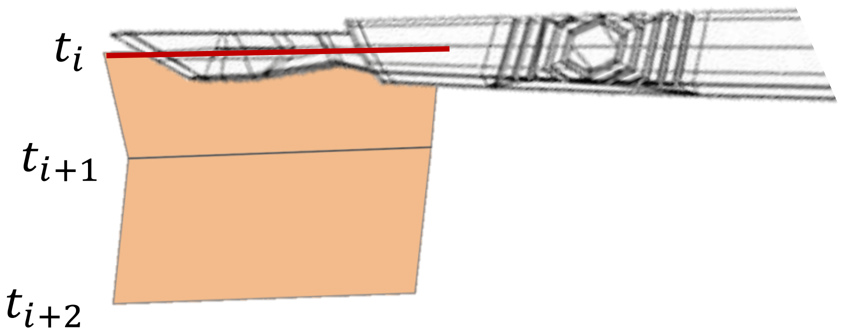


Object volumetric mesh

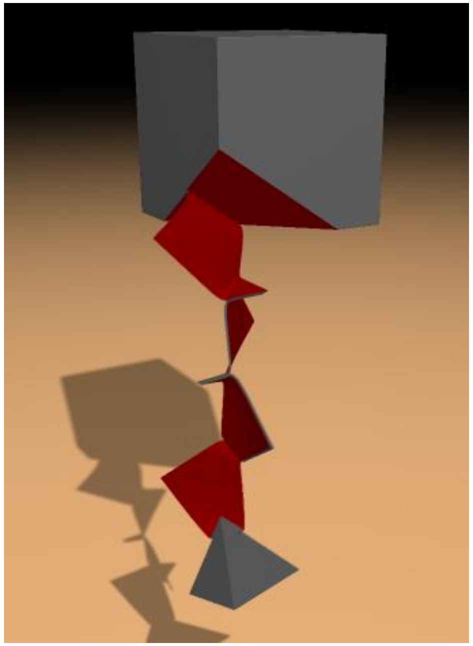
- Acceleration techniques
  - Bounding volume hierarchies
  - Breadth-first traversal of the volumetric mesh

Modeling of the Cutting Process

- Cutting surface generation
  - Swept surface of the cutting blade (interactive simulation)
  - Predefined cutting patterns (offline simulation)



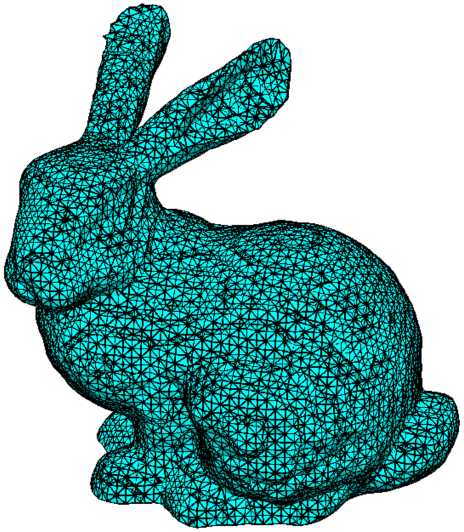
Swept surface



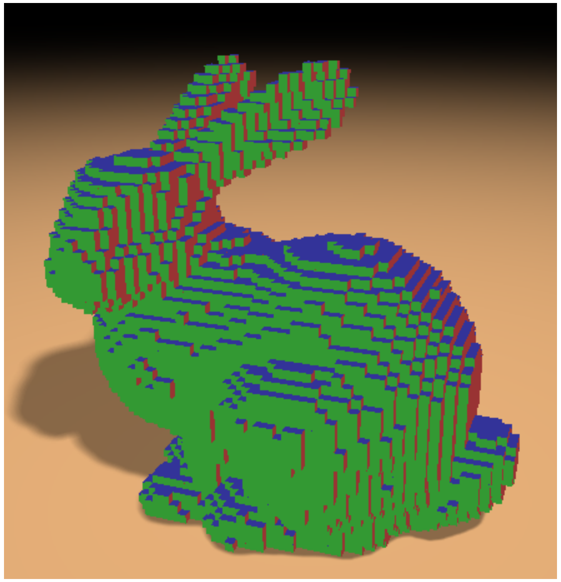
Cutting using a predefined pattern

Spatial Discretizations

- 2D: triangles, quadrangles, polygons
- 3D: tetrahedra, hexahedra, polyhedra



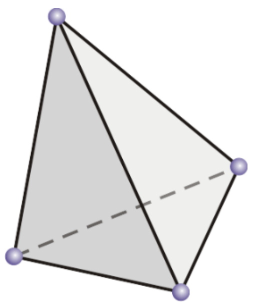
Tetrahedralized bunny model



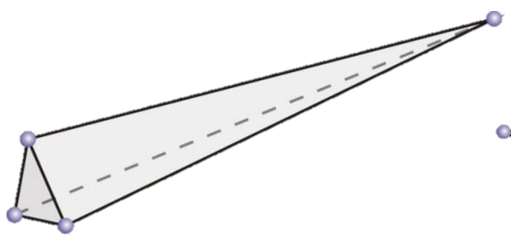
Hexahedralized bunny model

Tetrahedral Meshes

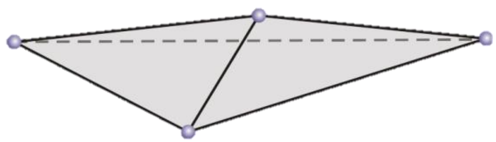
- Widely applied in computer graphics & engineering
- An initial tetrahedral discretization of the simulation domain can be generated
  - from surface meshes, medical images, level sets, et al.
  - by TetGen, LBIE-Mesher, et al.
- Challenge: avoiding ill-shaped meshes
  - Ill-shaped meshes lead to numerical instabilities
  - Mesh quality is ensured in the non-trivial initialization



Good-shaped



Ill-shaped, needle

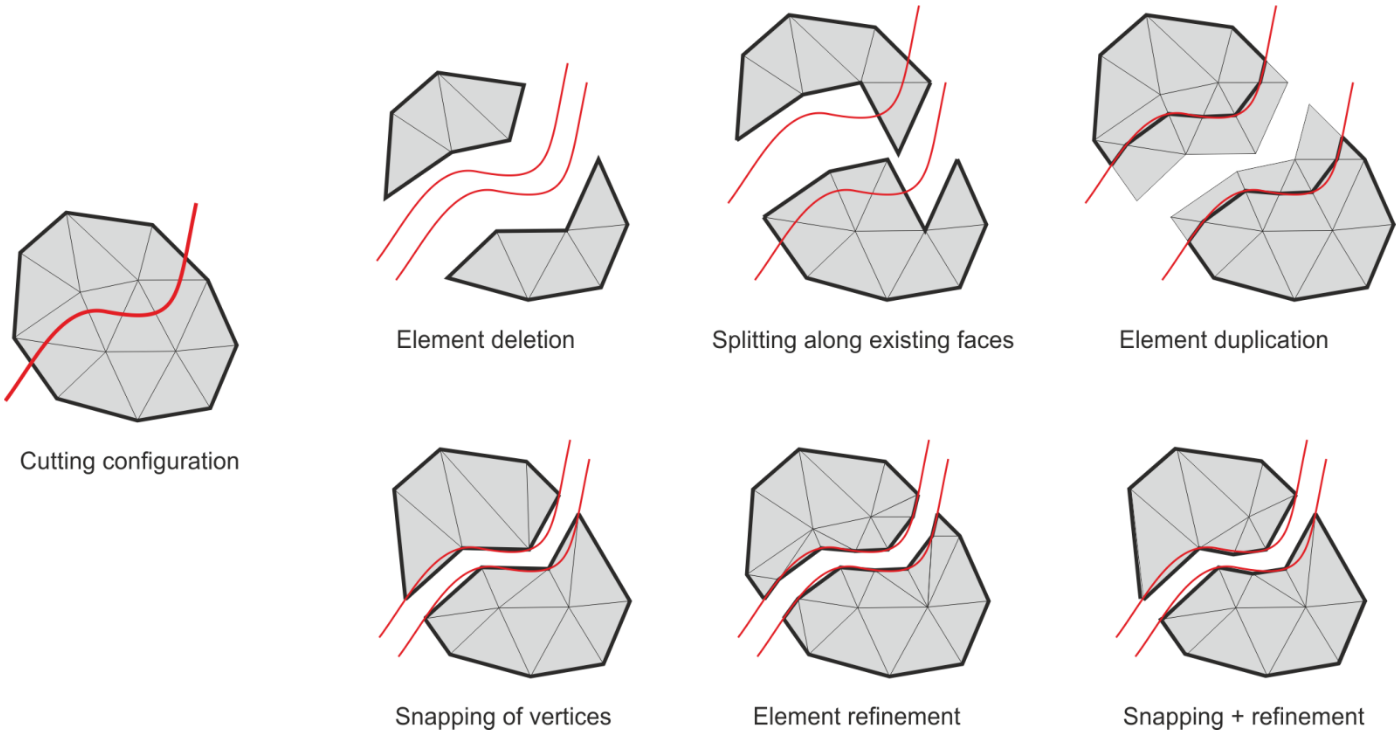


Ill-shaped, sliver

…

Tetrahedral Meshes

- Many techniques to model cuts into tetrahedral meshes

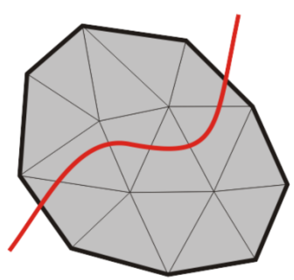


Techniques for modeling cuts in a tetrahedral mesh (a triangle mesh in 2D)

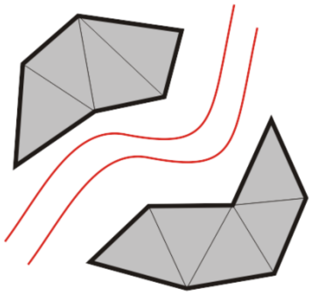
Cut Modeling without Creating New Elements

- Element deletion
  - Remove meshes that are touched by a cutting tool

➢ Simple, but result in a jagged surface and a loss of volume



Cutting configuration

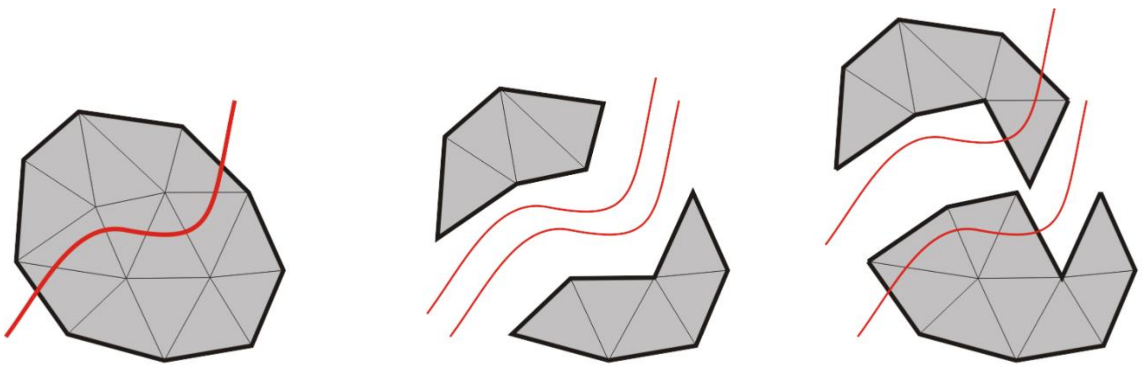


Element deletion

Cut Modeling without Creating New Elements

- Element deletion
- Splitting along existing faces

➢ Simple, but result in a jagged surface ~~and a loss of volume~~

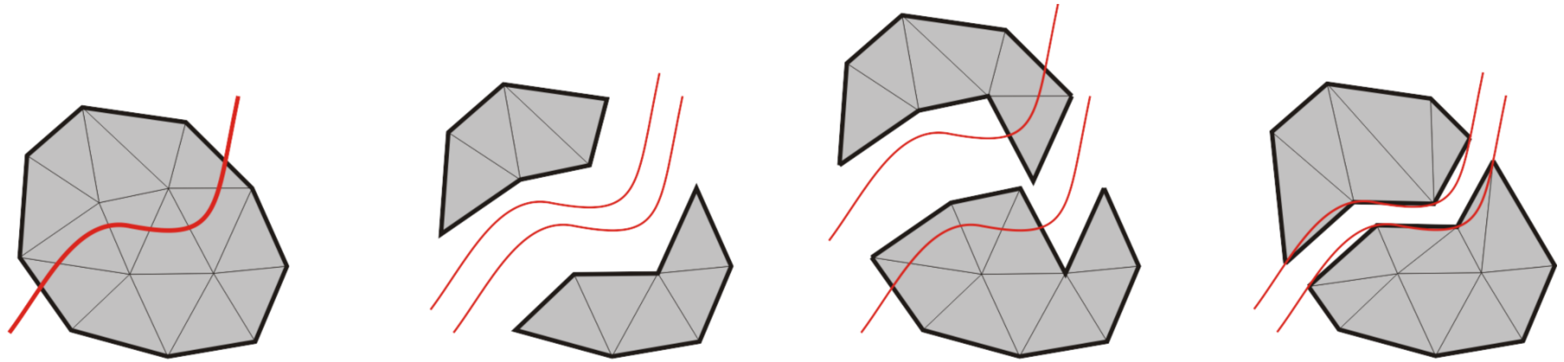


Cutting configuration | Element deletion | Splitting along existing faces

Cut Modeling without Creating New Elements

- Element deletion
- Splitting along existing faces
- Snapping of vertices
  - Snap vertices onto the cutting surface, i.e., positions altered
  - Then, split along faces

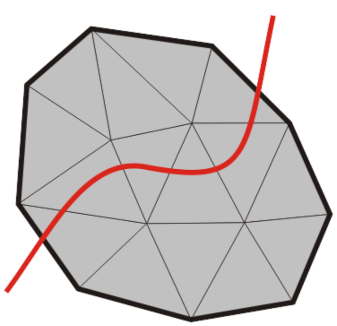
➢ Partially alleviate the jagged surface, but mesh quality cannot be ensured



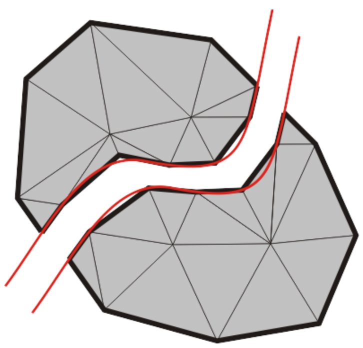
Cutting configuration | Element deletion | Splitting along existing faces | Snapping of vertices

Cut Modeling by Element Refinement

- Motivation: to accurately model a cut
- Solution: refine meshes along the cut
  - Split edges at the exact intersections
  - Create new, smaller meshes



Cutting configuration

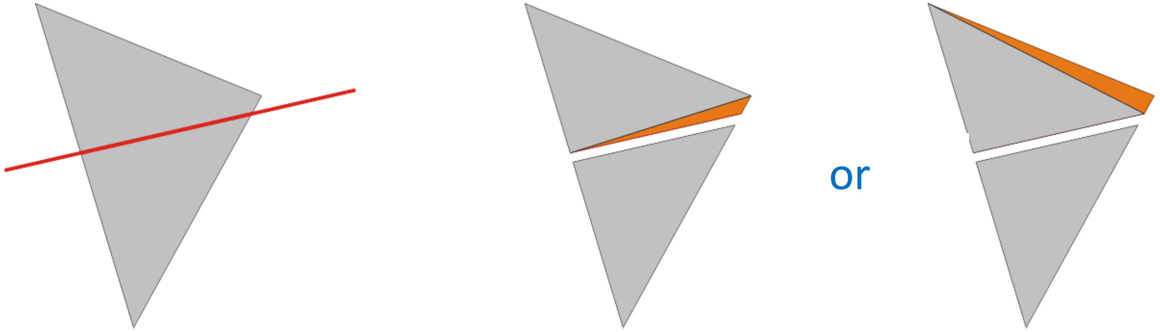


Element refinement

Cut Modeling by Element Refinement

- Motivation: to accurately model a cut
- Solution: refine meshes along the cut
  - Split edges at the exact intersections
  - Create new, smaller meshes

➢ Geometrically accurate, but easily lead to ill-shaped meshes
  - If the intersection is close to an initial vertex

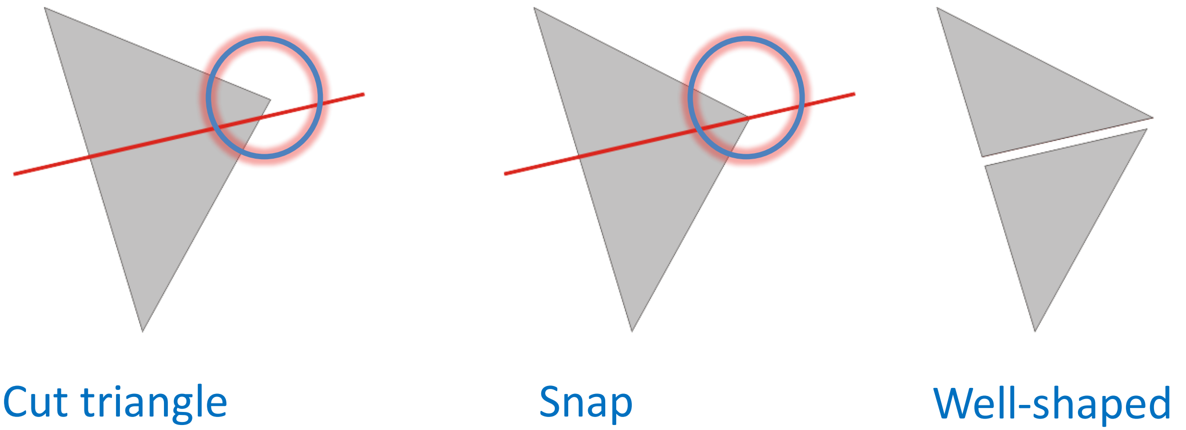


Cut triangle

Ill-shaped, needles

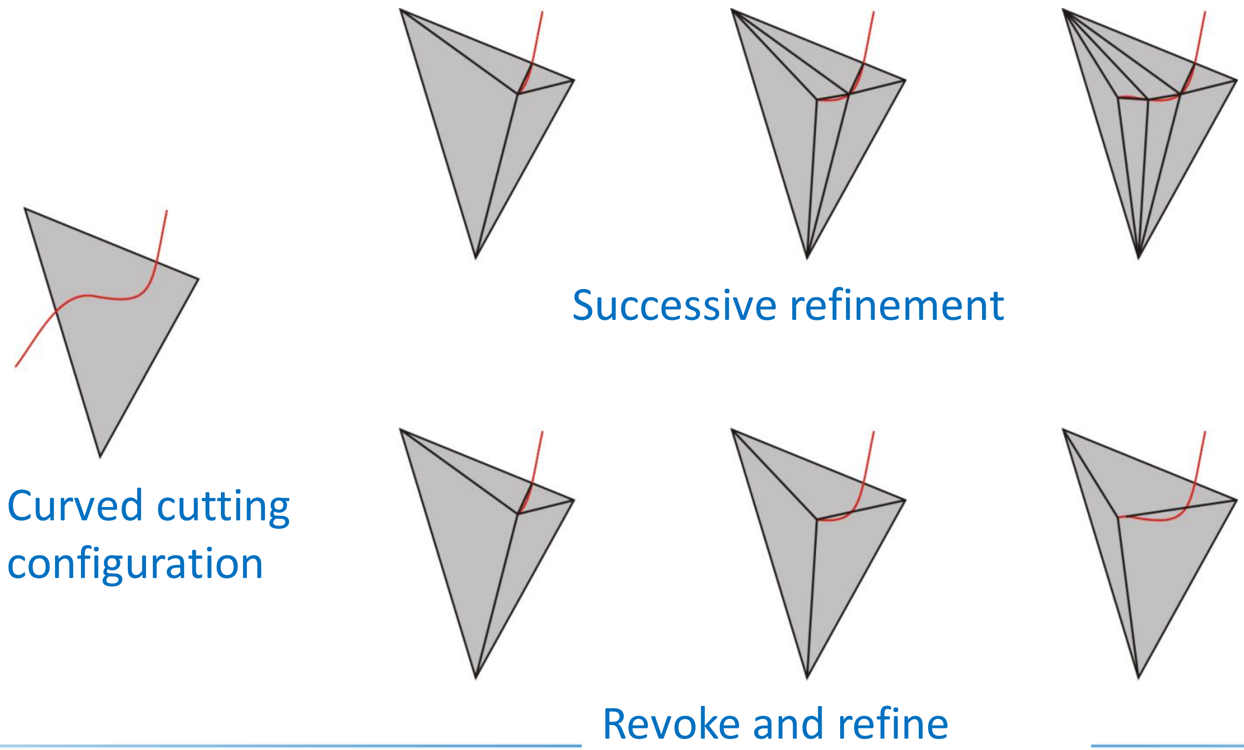
Cut Modeling by Element Refinement

- Motivation: to improve mesh quality
- Solution: a combination of snapping & refinement
  - Snap the vertex, if the intersection is close to it
  - Split the edge, otherwise



Cut Modeling by Element Refinement

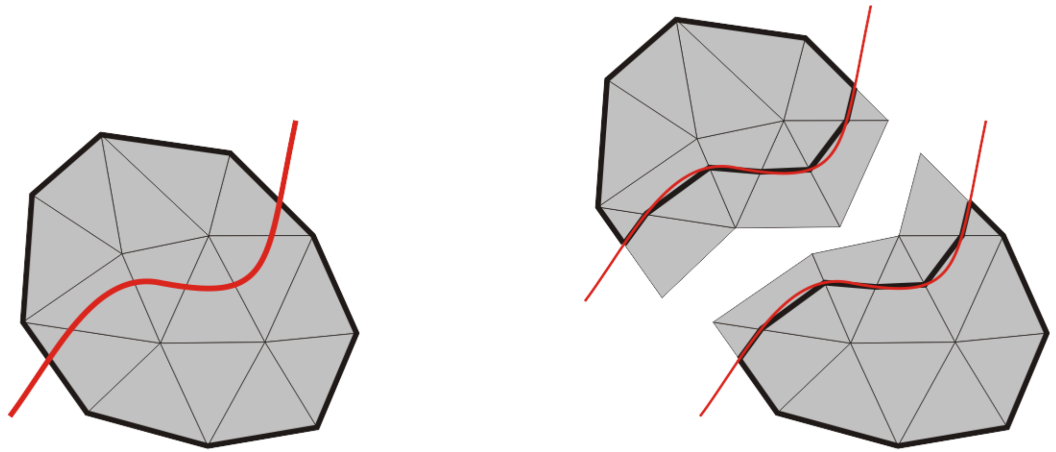
- Incremental, curved cutting path within one mesh
- Solutions:
  - Successive refinement
  - Revoke and refine



Cut Modeling by Element Duplication

- Motivation: to avoid ill-shaped elements

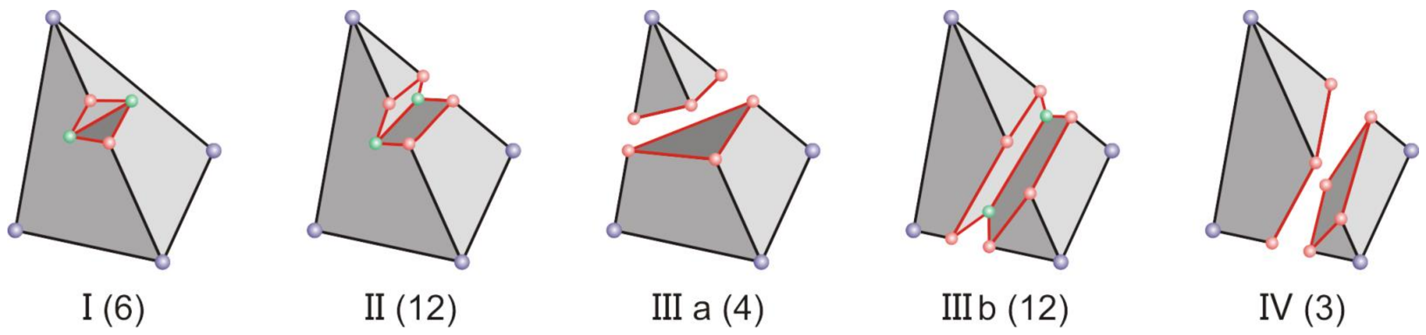
- Solution: duplicate the initial well-shaped elements
  - Create replicas of the elements that are cut
  - Embed material surfaces into a unique replica



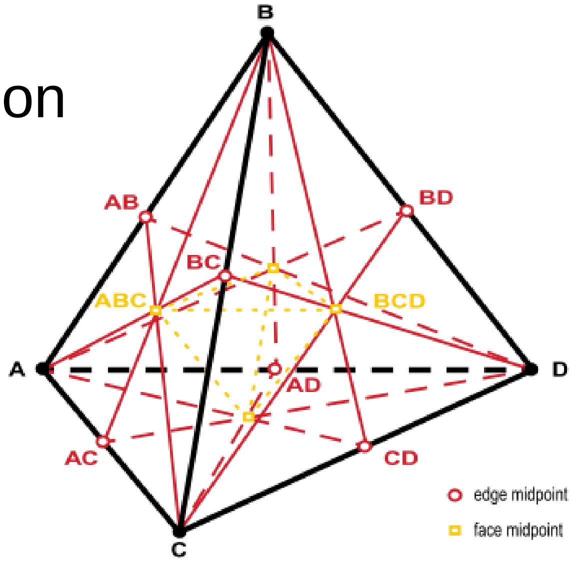
Cutting configuration   Element duplication

Tetrahedral decomposition

- Topological configurations of a cut tetrahedron



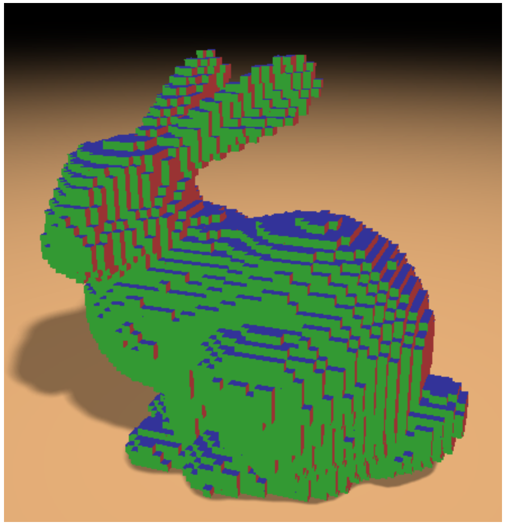
- Generic 1:17 tetrahedral decomposition
  - Add a vertex on each edge
  - Add a vertex on each triangle face
  - Exact placements decided by intersection tests



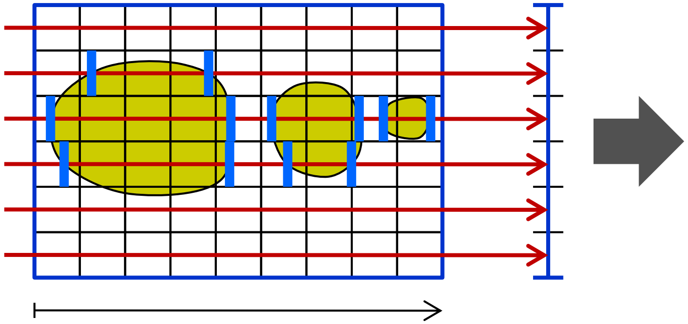
[Biielser et al. 1999]

Hexahedral Meshes

- Each element has a regular shape
- No worry about numerical instabilities!
- Generated from
  - medical images
  - polygonal surfaces by voxelization



Hexahedralized bunny model

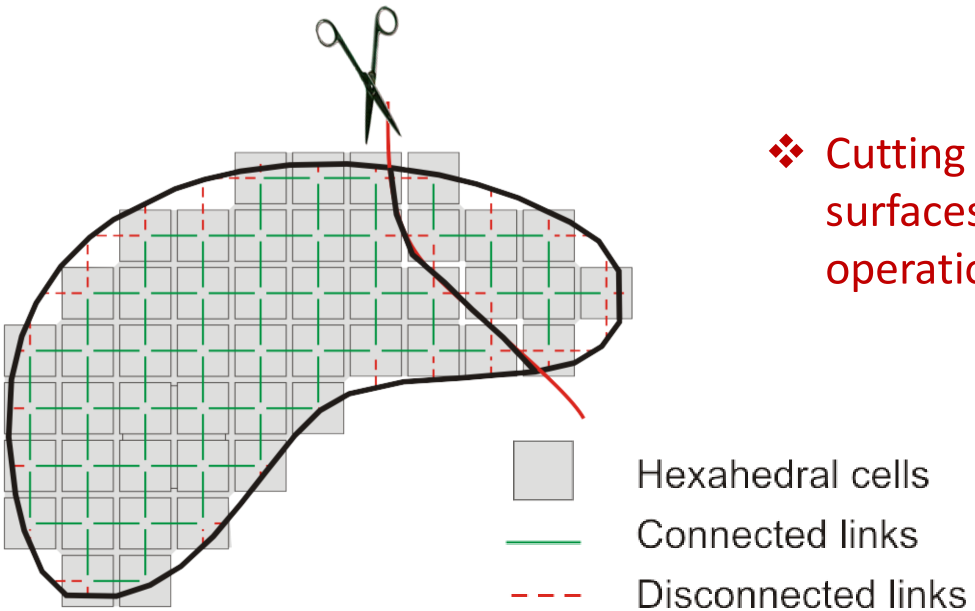


0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0
1	1	1	1	0	1	1	0	1
0	1	1	1	0	0	1	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

2D illustration of voxelization

Hexahedral Meshes - Volume Representation

- Linked volume
  - Decompose the object into a set of uniform hexadedra
  - Connect face-adjacent elements by links
  - Cutting: break the link between elements

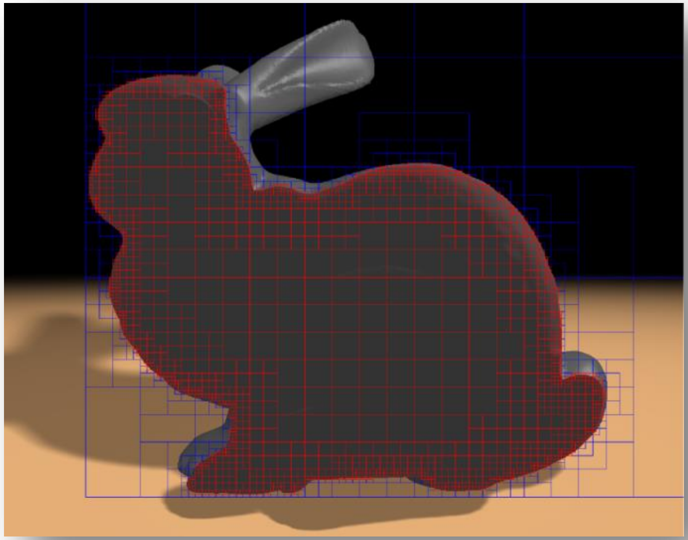


- Cutting surfaces and object boundary surfaces are both considered as cutting operations to break the links

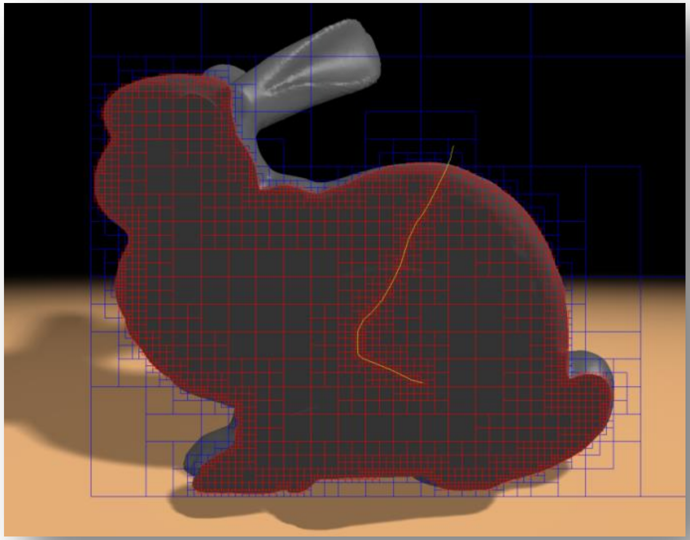
2D illustration of cutting on a linked volume

Hexahedral Meshes - Volume Representation

- Adaptive linked octree
  - Cutting: refine local elements, then break links
  - Regular 1:8 hexahedral decomposition
    - Efficient
    - No ill-shaped elements



Initial octree



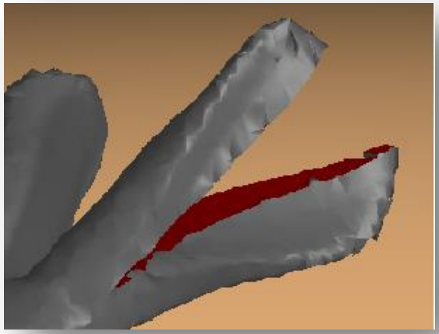
Refined octree

Hexahedral Meshes - Surface Representation

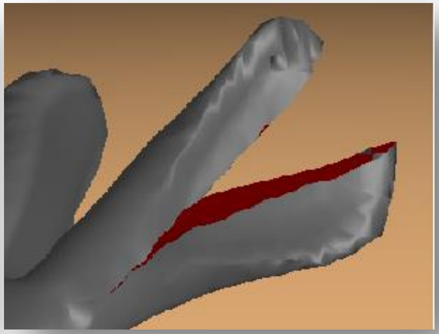
- Surface reconstruction methods
  - Marching cubes
  - Splitting cubes
  - Dual contouring

[Jeřábková et al. 2010]

Using marching cubes



Using splitting cubes

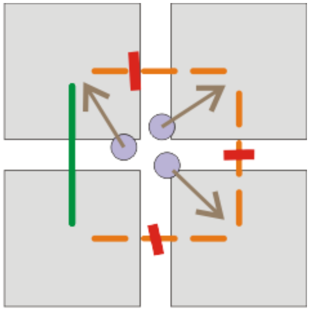


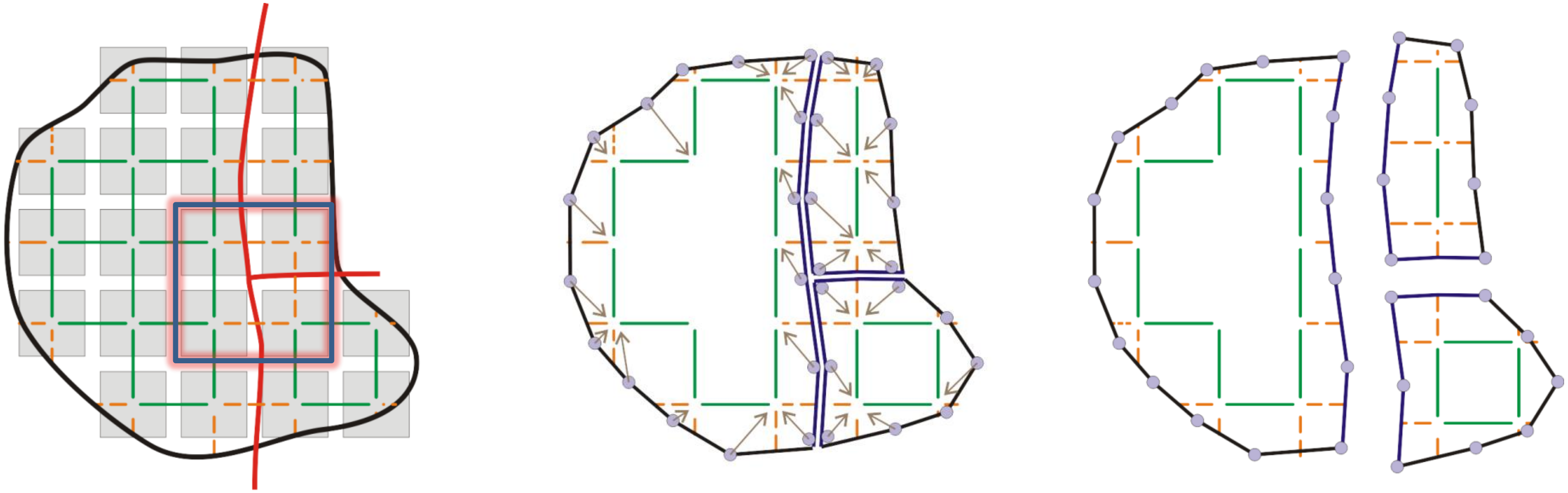
Using dual contouring

Surface reconstruction after cutting by different methods

Hexahedral Meshes - Surface Reconstruction

- Input: positions of intersection points & cutting normals
- Algorithm: (For each 2^3 block of elements)
  - Compute a surface vertex position which best matches all cuts
  - Duplicate the vertices as many times as the number of disconnected parts
  - Bind each replica to a volume element

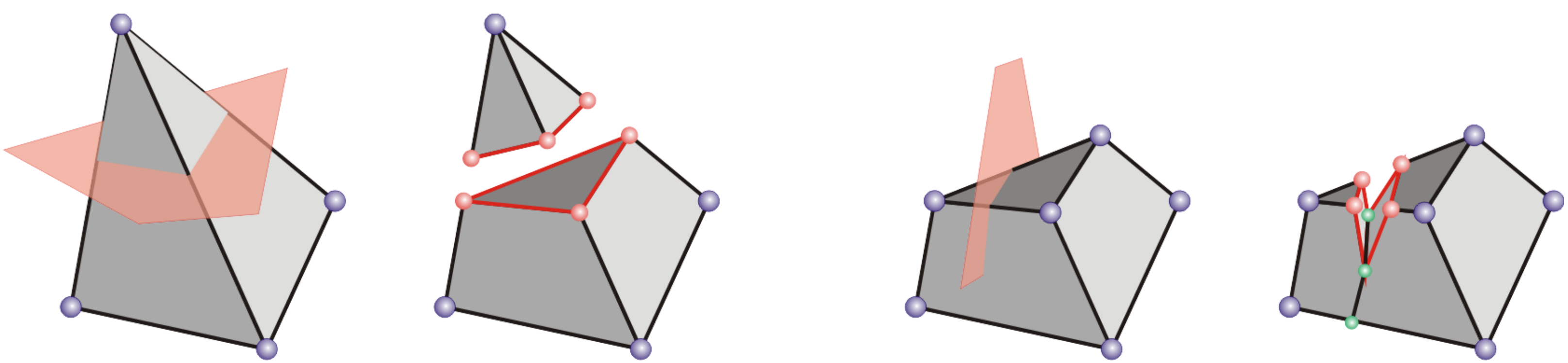




2D illustration of surface reconstruction

Polyhedral Meshes

- Flexible in representing shapes
  - Split the elements along a cutting plane
  - No further subdivision (e.g., tetrahedralization) is required
- Pros: no further subdivision is required
- Cons: ill-shaped elements needs to be avoided



A tetrahedron -> A small tetrahedron & A triangular prism

A tetrahedron -> Two small tetrahedra

Discussion on Discretizations

- Tetrahedral & polyhedral meshes
  - Pros: flexibility in shape modeling, directly renderable surfaces
  - Cons: ill-shaped elements
  - Methods:
    - element deletion, splitting along existing faces, element duplication, snapping of vertices, element refinement, snapping + refinement
- Hexahedral meshes
  - Pros: efficiency wrt. subdivision and solvers, stability
  - Cons: a separate surface is needed
  - Methods:
    - (adaptive) linked volume, surface reconstruction

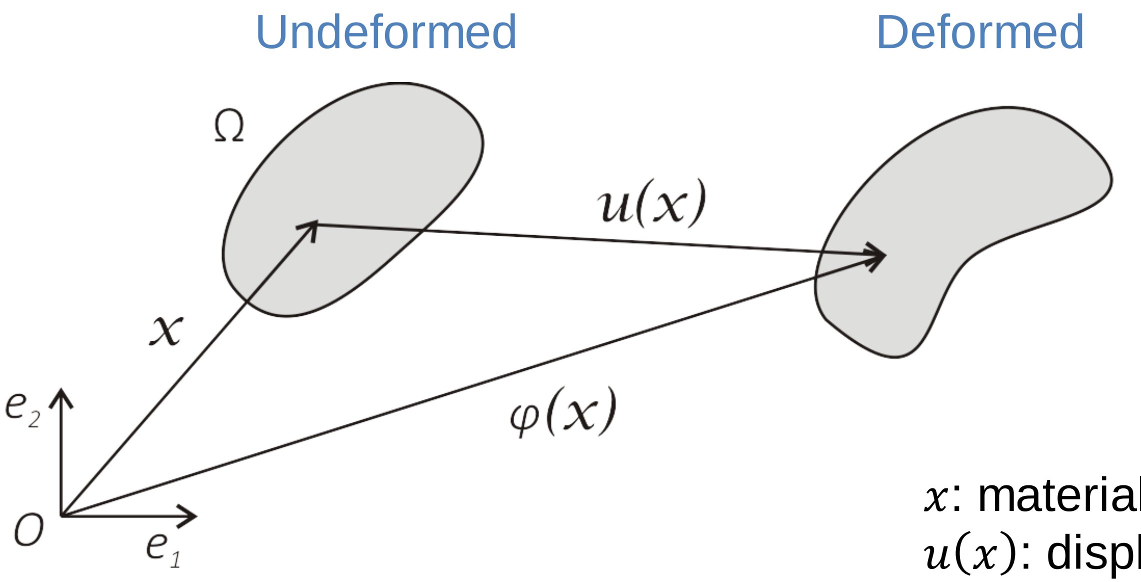
Outline

follows the structure of the report

- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation for Virtual Cutting
  - Extended FEM
  - Composite FEM
  - Polyhedral FEM
  - Discussion on FEMs
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Physically-based Deformation Models

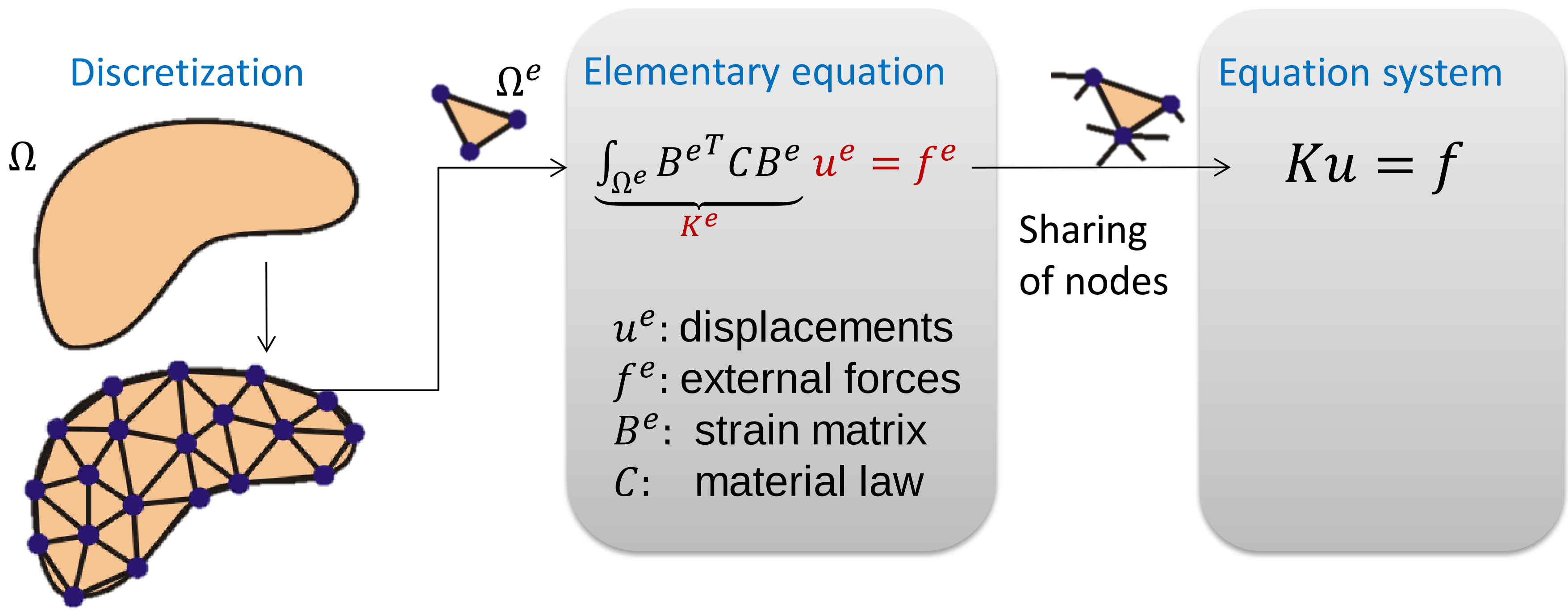
- Compute the object's deformation due to external forces
  - Introduced to computer graphics by Terzopoulos et al. 1987
  - Surveyed in STAR by Nealen et al. 2006
- Finite element methods (FEM), meshfree methods, mass-spring systems, etc.



x: material coordinates
$u(x)$: displacement field
$\varphi(x)$: deformation field

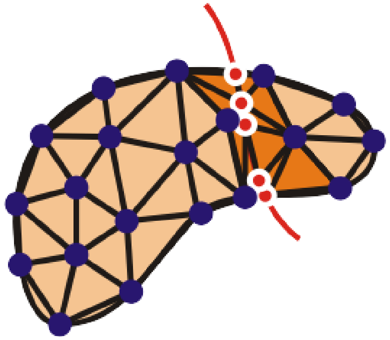
Recap: Finite Element Simulation of Elasticity

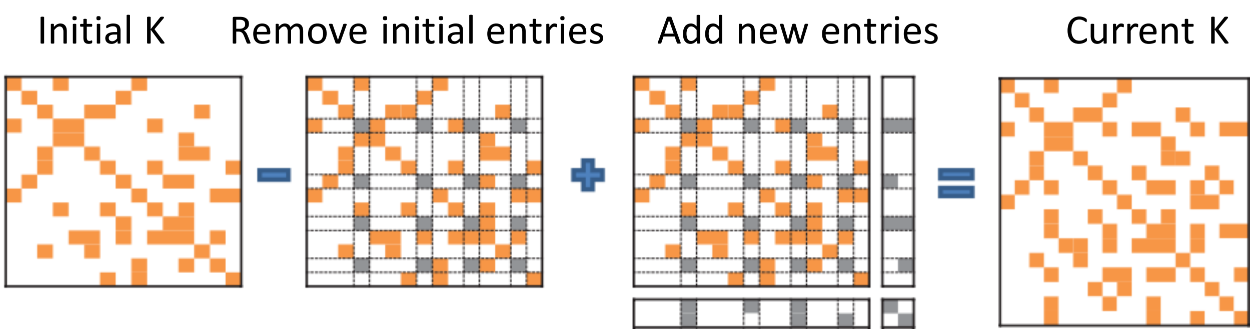
1) Discretize the object into elements
2) Build elementary equations $K^e u^e = f^e$
3) Assemble a linear system of equations $Ku = f$
4) Solve for the displacement field u



Virtual Cutting Using the Standard FEM

1) Split elements which are touched by the scalpel
2) Re-build elementary equations $K^e u^e = f^e$
3) Re-assemble a linear system of equations $Ku = f$
   - Remove entries of the deleted initial elements
   - Add entries of the split new elements
4) Solve for the displacement field u



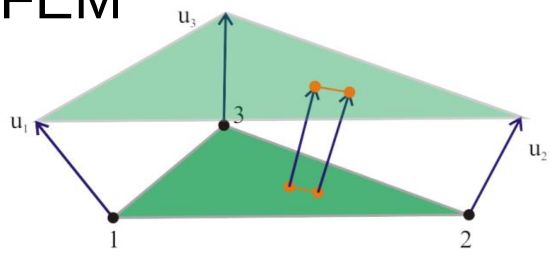


Re-assemble the stiffness matrix [Courtecuisse et al 2014]

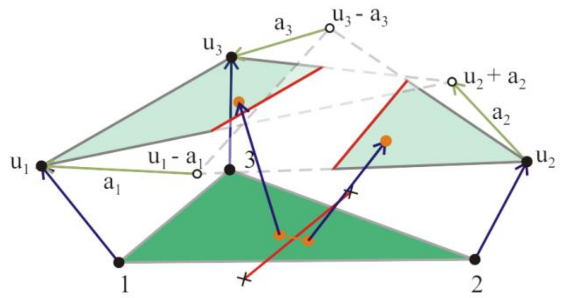
The Extended Finite Element Method (XFEM)

- Model material discontinuities by enriching the basis functions of the solution space [Belytschko et al. 1999]
  - Adapting basis functions instead of modifying the meshes

- Displacement field $u(x)$ in the standard FEM
- $u(x) = \Phi^e(x)\, u^e$
  - $\Phi^e(x)$: shape matrix
  - u^e: displacement vector at nodes

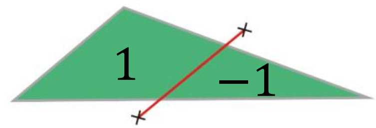


- Displacement field $u(x)$ in the extended FEM
- $u(x) = \Phi^e(x)\, u^e + \Psi^e(x)\Phi^e(x)\, a^e$
  - $\Psi^e(x)$: shape enrichment matrix
  - a^e: added displacement vector at nodes

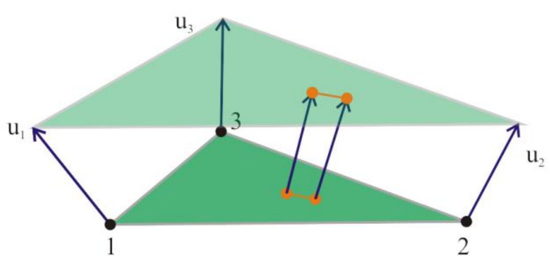


XFEM – Discontinuous Enrichment Function

- $u(x) = \Phi^e(x)\, u^e + \Psi^e(x)\Phi^e(x)\, a^e$
- Shifted enrichment function
  - $\psi_i^e(x) = \frac{H(x)-H(x_i)}{2}$
  - $H(x) = \begin{cases} 1, & \text{if } x \text{ is on the cut's left side;} \\ -1, & \text{if } x \text{ is on the cut's right side.} \end{cases}$



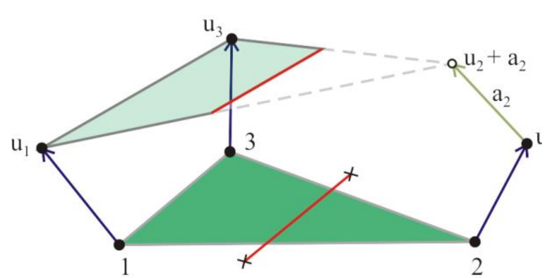
Heaviside function $H(x)$



Both left and right sides:

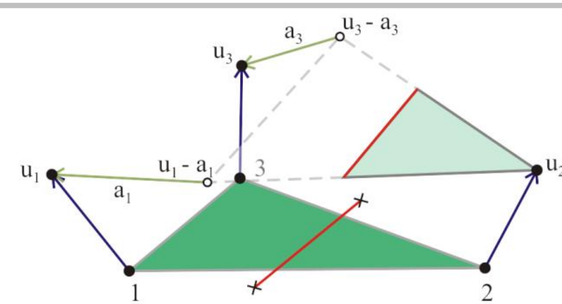
$$u(x) = \Phi^e(x)\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

Standard FEM



Left side of the triangle:

$$u(x) = \Phi^e(x)\begin{pmatrix} u_1 \\ u_2 + a_2 \\ u_3 \end{pmatrix}$$



Right side of the triangle:

$$u(x) = \Phi^e(x)\begin{pmatrix} u_1 - a_1 \\ u_2 \\ u_3 - a_3 \end{pmatrix}$$

Extended FEM

XFEM - Stiffness Matrices

- Standard stiffness matrix $K^e := \int_{\Omega^e} {B^e}^T C B^e$
  - Material law C relates strain to stress $\sigma = C : \epsilon$
  - Strain matrix $B^e = \left(B_1^e, \dots, B_{n_v}^e\right)$

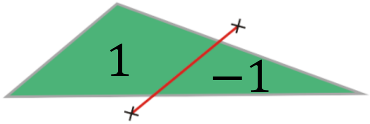
- Enriched stiffness matrix ${}^xK^e = \int_{\Omega^e} ({}^xB^e)^T \, C \; {}^xB^e dx$
- ${}^xB^e = \left(B_1^e, \dots, B_{n_v}^e, \;\; \psi_1^e B_1^e, \dots, \psi_{n_v}^e B_{n_v}^e\right)$
- ${}^xK^e = \begin{pmatrix} K^{e,uu} & K^{e,ua} \\ K^{e,au} & K^{e,aa} \end{pmatrix}$

XFEM – Detailed Cutting of Shells

- Store enrichment function as a 2D texture

Enrichment texture within a quad mesh

Simulation result



Heaviside function $H(x)$

$$H(x) = \begin{cases} 1, & \text{on left} \\ -1, & \text{on right} \end{cases}$$

[Kaufmann et al 2009]

XFEM – Detailed Cutting of Shells

- Store enrichment function as a 2D texture

Enrichment texture
within a quad mesh

Simulation result

Multiple cuts

[Kaufmann et al 2009]

XFEM – Detailed Cutting of Shells

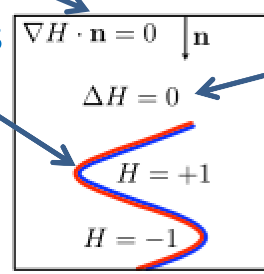
- Store enrichment function as a 2D texture
- Employ harmonic enrichment function for partial cuts

Enrichment texture within a quad mesh

Simulation result

Enrichment texture of a partial cut

Boundary conditions



Laplace eq.

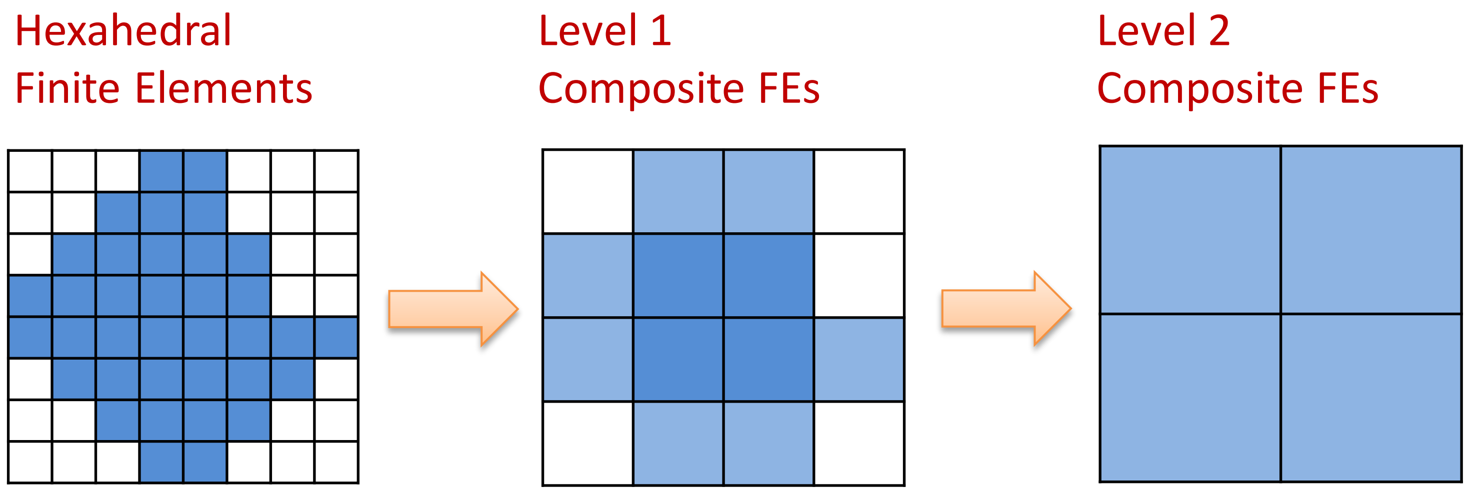
Harmonic enrichments $H(x)$

Simulation result

[Kaufmann et al 2009]

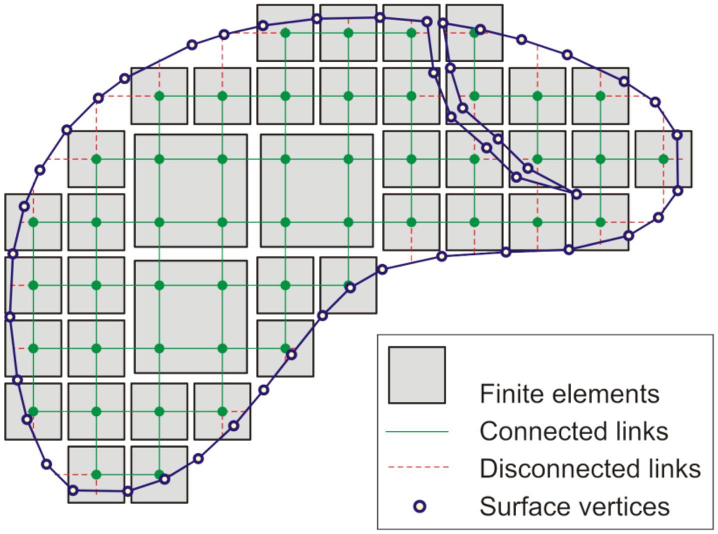
The Composite Finite Element Method (CFEM)

- Approximate a high resolution finite element discretization by a small set of coarser elements [Hackbusch & Sauter 1997]
  - Reduce the number of simulation DOFs
  - Also used for: construct a grid hierarchy for the multigrid solver

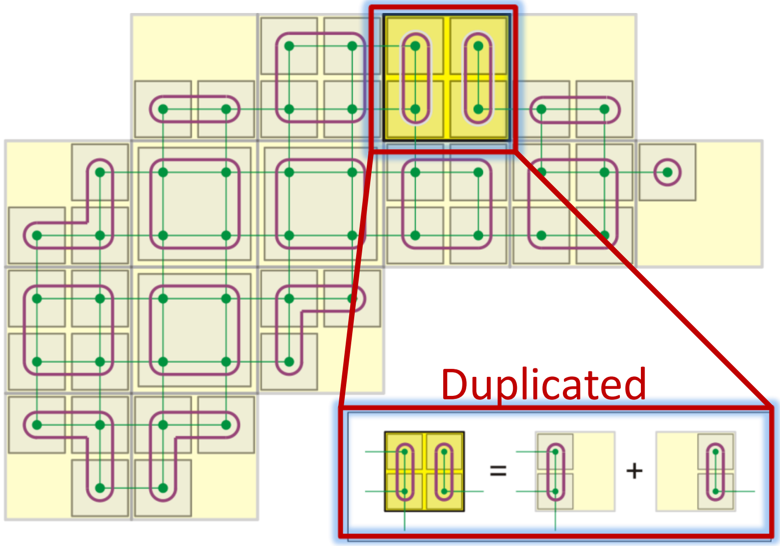


CFEM – Geometrical & Topological Composition

- Duplicated elements: Each connected part is merged to one independent element
  - Located at the same place in the reference configuration
  - But have different topology connections



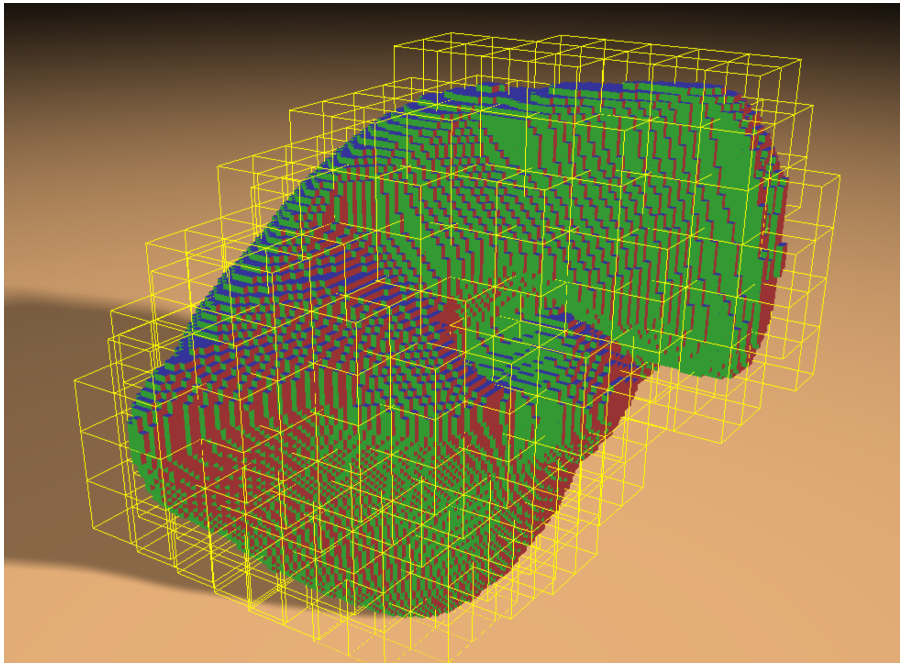
Linked octree representation



Composite finite element

CFEM – Geometrical & Topological Composition

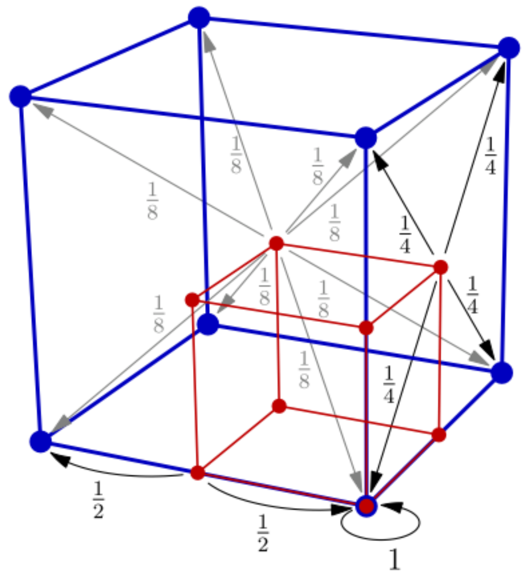
- Duplicated elements: Each connected part is merged to one independent element
  - Located at the same place in the reference configuration
  - But have different topology connections
- Iteratively merge blocks of 2^3 elements into 1 element



Fine resolution: 82×83×100
Composition level: 3 (8^3->1)

CFEM – Numerical Composition

- Displacement interpolation
  - composite elements → fine hexahedra
  - $u = I\,\tilde{u}$
- Stiffness matrix assembly
  - fine hexahedra → composite elements
  - $\widetilde{K} = I^{\mathrm{T}} K I$

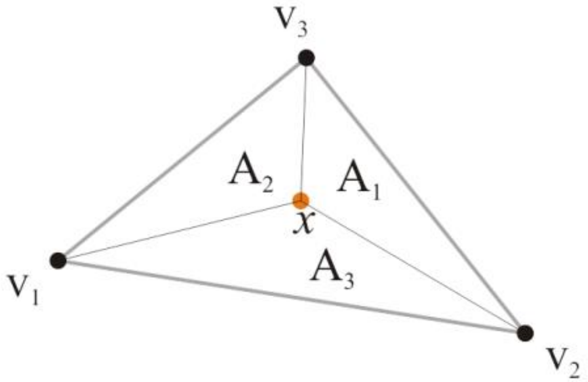


- $\widetilde{K}^{c}_{mn} = \sum_{e \text{ in } c} \sum_{i=1}^{8} \sum_{j=1}^{8} w^{c\to e}_{m\to i} w^{c\to e}_{n\to j} K^{e}_{ij}, \quad m,n = 1,\dots,8$
- $w^{c\to e}_{m\to i} = \left(1 - \frac{|x^c_m - x^e_j|}{s^c}\right)\left(1 - \frac{|y^c_m - y^e_j|}{s^c}\right)\left(1 - \frac{|z^c_m - z^e_j|}{s^c}\right)$

The Polyhedral Finite Element Method (PFEM)

- Directly evaluate deformation on general polyhedra [Wicke et al. 2007]
  - Tetrahedralization/hexahedralization process is avoided

- Shape functions: $u(x) = \sum_{i=1}^{n_v} \phi_i(x)\, u_i$
  - Tetrahedron: barycentric interpolation
  - Hexahedron: tri-linear interpolation
  - Polyhedron: ??



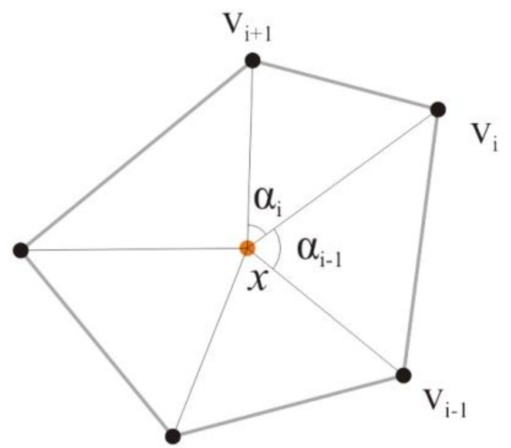
$$\phi_i(x) = \frac{A_i}{\sum_{j=1}^{n_v} A_j} u_i,$$

$$A_i = A(x, v_{i-1}, v_{i+1})$$

Barycentric interpolation for a triangle

PFEM – Shape Functions

- Mean value interpolation function
  - Generalization of barycentric interpolation to convex polyhedra
- Shape functions: $u(x) = \sum_{i=1}^{n_v} \phi_i(x)\, u_i$
  - Kronecker delta property: $\phi_i(x_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$
  - Completeness: $\sum_{i=1}^{n_v} \phi_i(x) = 1$



$$\phi_i(x) = \frac{w_i}{\sum_{j=1}^{n_v} w_j}$$

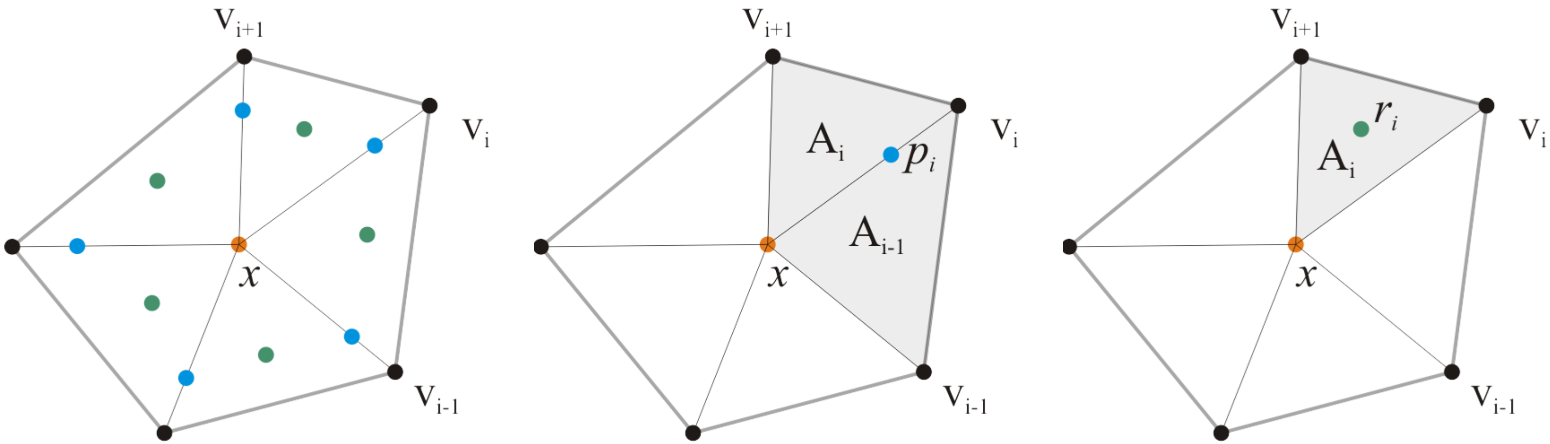
$$w_i = \frac{\tan\left(\alpha_{i-1}/2\right) + \tan\left(\alpha_i/2\right)}{||v_i - x||}$$

Mean value interpolation for a polygon

PFEM – Stiffness Matrices

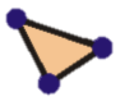
- Stiffness matrix $K^e := \int_{\Omega^e} {B^e}^T C B^e$
  - Analytical integration over general polyhedra is non-trivial
  - Approximated by numerical integration at a few samples
- $K^e = \sum_i \frac{\mu_i^e}{2} (B^e(p_i))^{\mathrm{T}}\, C\, B^e(p_i) + \sum_i \frac{\kappa_i^e}{2} (B^e(r_i))^{\mathrm{T}}\, C\, B^e(r_i)$

$\mu_i^e = \frac{A_{i-1} + A_i}{2A^e}$ $\kappa_i^e = \frac{A_i}{A^e}$



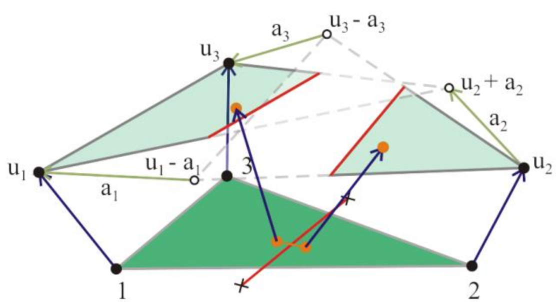
Discussion on FEMs

- Standard FEM
  - Each spatial mesh maps to one specific computational finite element

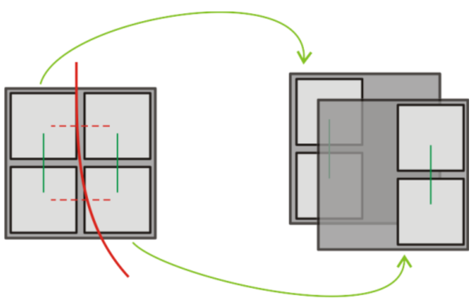


$$K^e u^e = f^e$$

- Extended FEM, composite FEM
  - Disconnected spatial mesh corresponds to multiple, duplicated simulation DOFs



$$\begin{pmatrix} K^{e,uu} & K^{e,ua} \\ K^{e,au} & K^{e,aa} \end{pmatrix} \begin{pmatrix} u^e \\ a^e \end{pmatrix} = \begin{pmatrix} f^e \\ 0 \end{pmatrix}$$



Extended FEM

Composite FEM

Outline

follows the structure of the report

- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation for Virtual Cutting
- Numerical Solvers
  - Direct solvers
  - Iterative solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Numerical Solvers

- Implicit time integration leads to a linear system of equations
$$Ax = b$$
  - when using the linear strain tensor and a linear material model

- A is a sparse, symmetric, positive definite matrix

- Update of the system matrix A required …
  - due to adaptation of the finite element model (cutting)
  - in every time step, when using the corotational strain formulation
  - Requires re-initialization of the solver

Direct Solvers

- Obtain exact solution in a finite number of steps

- Matrix inversion: $b = A^{-1}x$ $(A \in \mathbb{R}^{n \times n})$
  - Computing time $O(n^3)$ (initialization) and $O(n^2)$ (solve)
  - Memory $O(n^2)$
  - Only feasible for (very) small n
  - Incremental update via Sherman-Morrison-Woodbury formulae
    - $(A - UV^T)^{-1} = A^{-1} + A^{-1}U(E - V^T A^{-1} U)^{-1} V^T A^{-1}$
    - Update can be restructured to be in $O(n)$ under certain assumptions considering the number of non-zero entries [Zhong et al. 2005]

Direct Solvers

- Cholesky factorization: $A = LL^T$ for a spd matrix A

$$L \underbrace{L^T x}_{y :=} = b$$
$$Ly = b$$
$$L^T x = y$$

  - Better constant factors than matrix inversion
  - Can also be incrementally updated [Turkiyyah et al. 2009]

Iterative Solvers

- Successively compute approximations x_m to the solution x

$$x = \lim_{m \to \infty} x_m$$

- Allows for balancing speed and accuracy
  - Monitor norm of residual $r_m = b - Ax_m$
  - Stop if residual reduction $\frac{\|r_m\|_2}{\|r_0\|_2} \leq \tau$ for given threshold τ

Iterative Solvers

- Conjugate Gradient Method

$$Ax = b \quad \Leftrightarrow \quad \underbrace{\frac{1}{2}x^T A x - b^T x}_{F(x):=} \rightarrow min \qquad \text{for spd matrix } A$$

  - F has a single, global minimum (paraboloid)
  - Iterative search for minimum:

$$x_{m+1} = x_m + \lambda_m p_m$$

$$p_m = -\nabla F(x_m) + \sum_{j=0}^{m-1} \alpha_j p_j \qquad p_i^T A p_j = 0 \text{ for } i \neq j$$

    - Problem-adapting
    - x_m minimizes F on affine subspace of continuously increasing dimension
  - Requires matrix-vector products and dot products
  - Efficient parallelization using OpenMP [Chentanez et al. 2009] or CUDA [Courtecuisse et al. 2010]

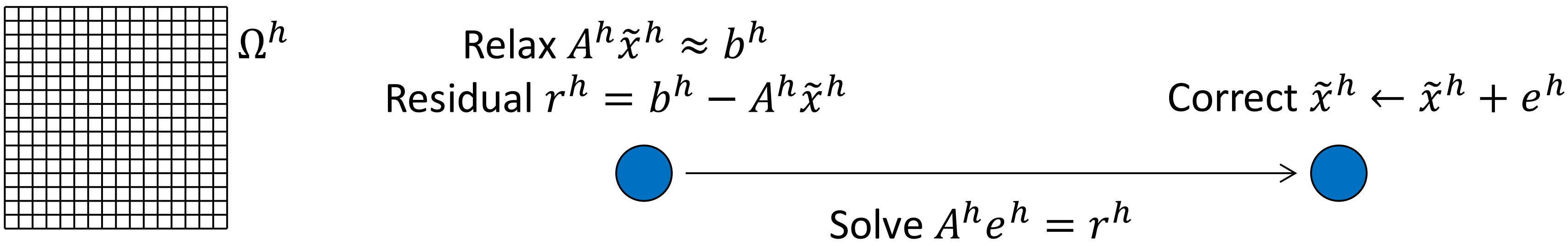
Iterative Solvers

- So far: “Blackbox” solvers

- More advanced solvers: Geometric multigrid solvers
  - Basic relaxation schemes (Jacobi, Gauss-Seidel) only reduce high-frequency error components effectively
  - Consider the problem on a hierarchy of successively coarser grids
  - Reduce lower-frequency error components on coarser grids (where they appear at a higher frequency)

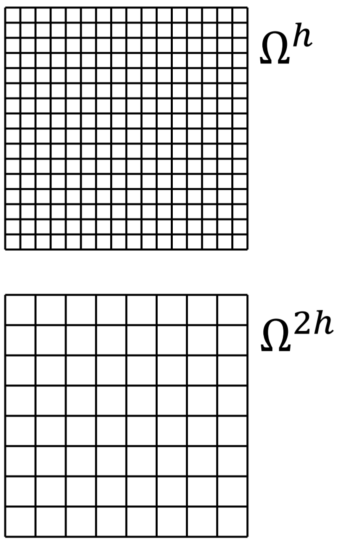
Geometric Multigrid

- Solve $A^h x^h = b^h$, current approximate solution $\tilde{x}^h$



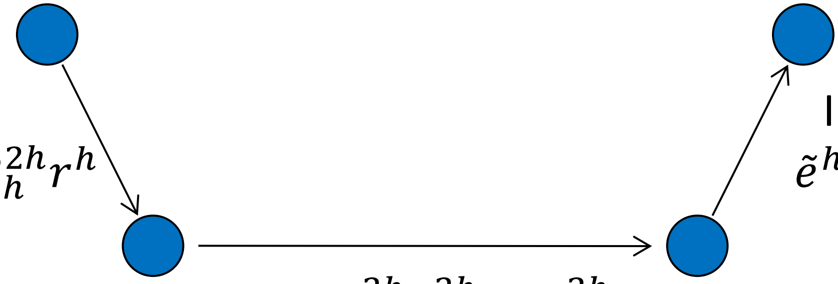
Geometric Multigrid

- Solve $A^h x^h = b^h$, current approximate solution $\tilde{x}^h$



Relax $A^h \tilde{x}^h \approx b^h$ (Pre-smoothing)
Residual $r^h = b^h - A^h \tilde{x}^h$

Relax $A^h \tilde{x}^h \approx b^h$ (Post-smoothing)
Correct $\tilde{x}^h \leftarrow \tilde{x}^h + \tilde{e}^h$ (Coarse Grid Corr.)



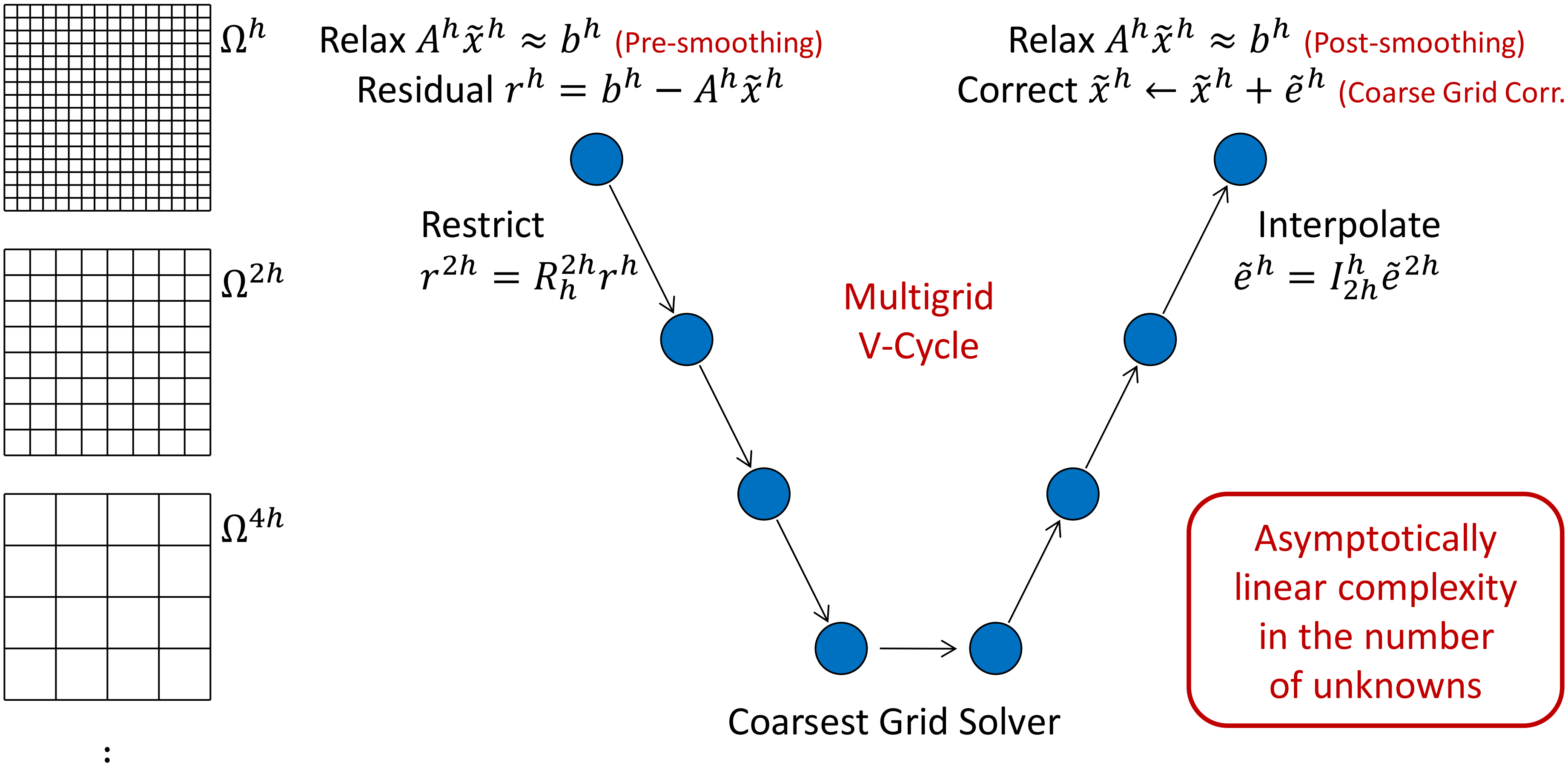
Restrict
$r^{2h} = R_h^{2h} r^h$

Interpolate
$\tilde{e}^h = I_{2h}^h e^{2h}$

Solve $A^{2h} e^{2h} = r^{2h}$

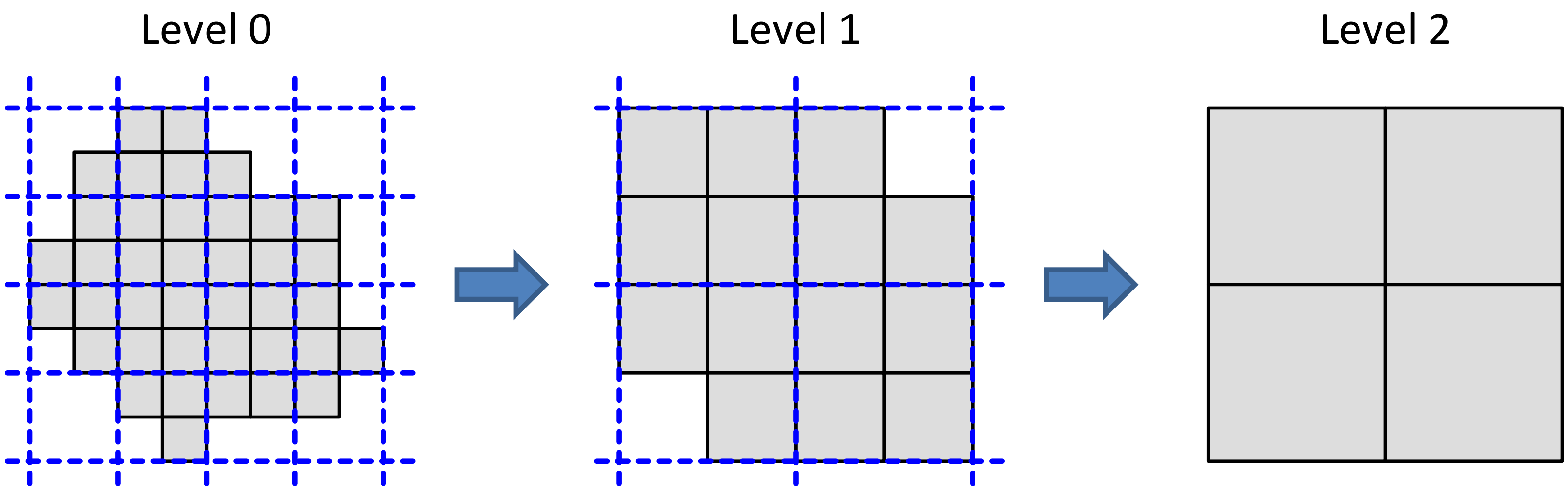
Geometric Multigrid

- Solve $A^h x^h = b^h$, current approximate solution v^h



Multigrid Hierarchy Construction

- (Semi-)Regular hexahedral grids

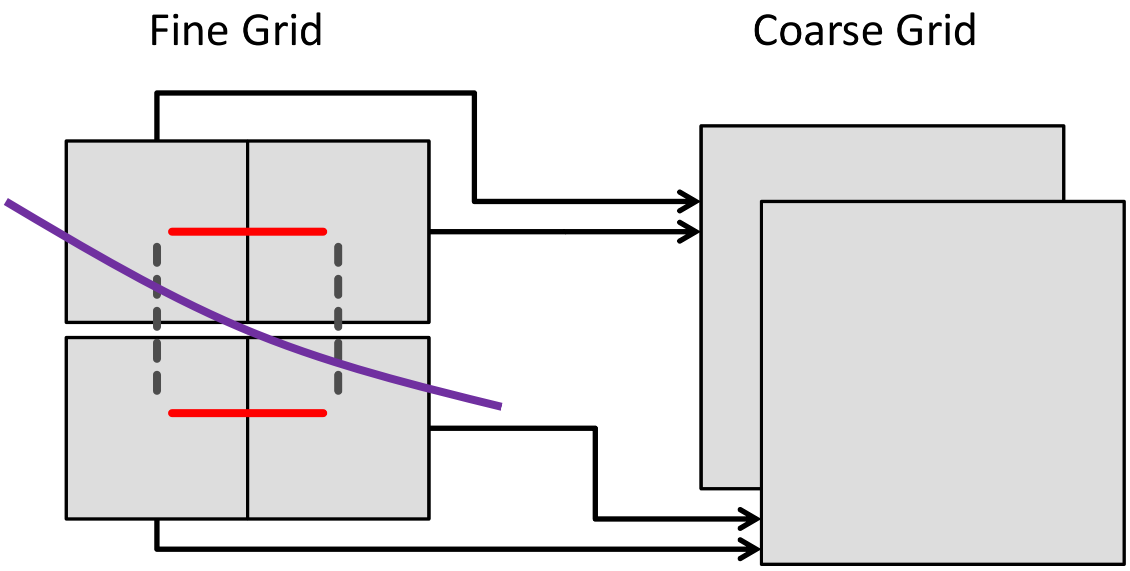


  - Blocks of 2^3 cells are merged into coarse grid cells of double size
  - A cell is created if it covers at least one cell on the finer level
    - Coarser cells might be only partially filled [Liehr et al. 2009]

- Difficult for unstructured grids

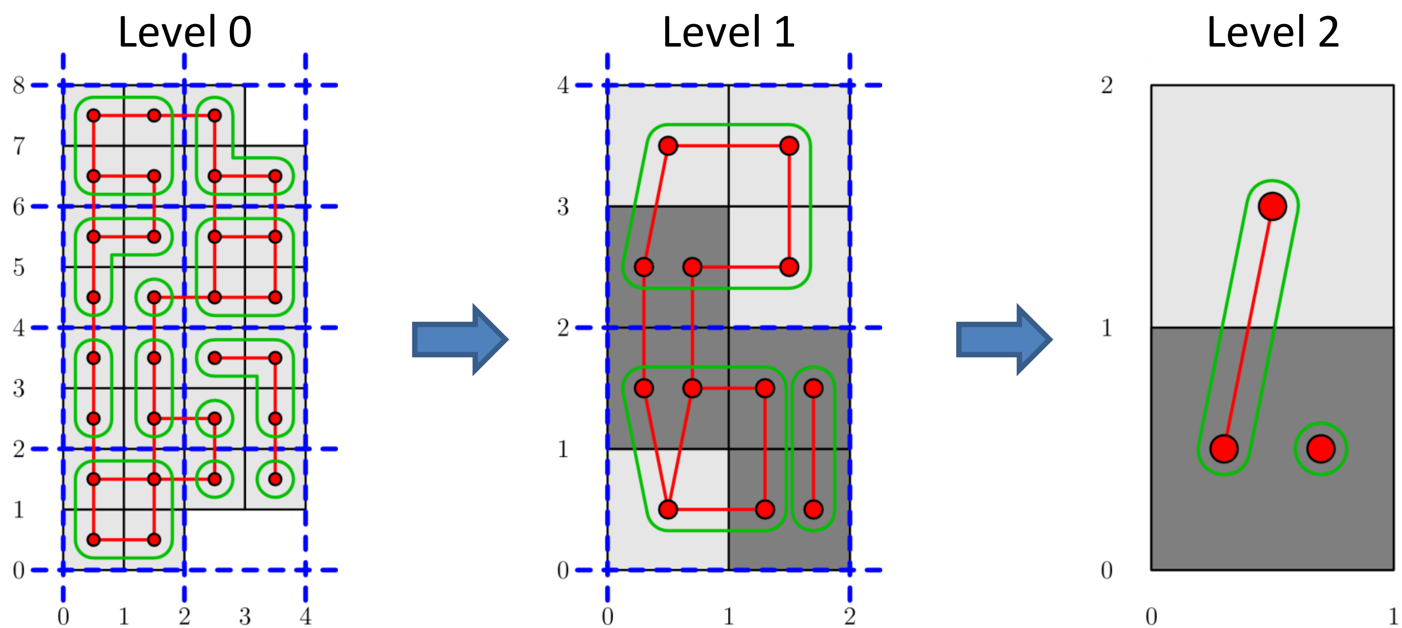
Multigrid with Cuts

- Representation of complicated topologies on the coarse grids
  - Physically disconnected parts should be represented by individual coarse grid cells
  - Duplication of cells on the coarse grids [Aftosmis et al. 2000]
  - Graph-based hierarchy construction analogous to composite elements



Multigrid with Cuts

- Construction of multigrid hierarchy using an undirected graph representation

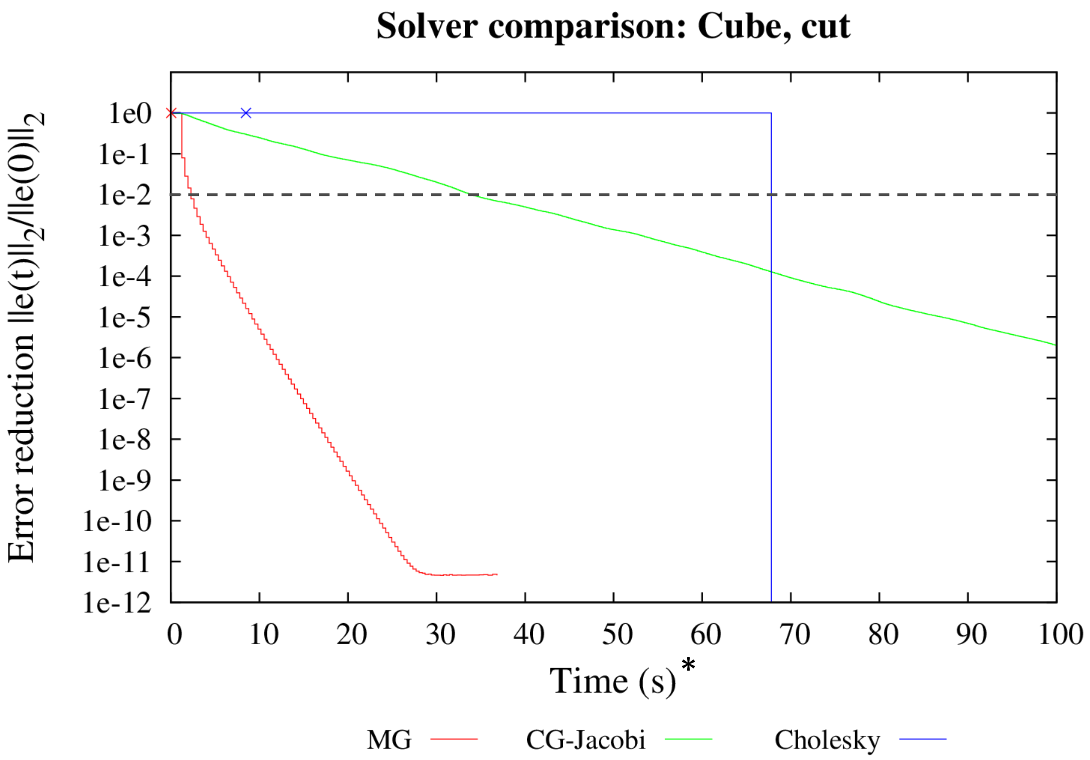


[Dick et al. 2011]

  – Works equally well for an adaptive octree grid

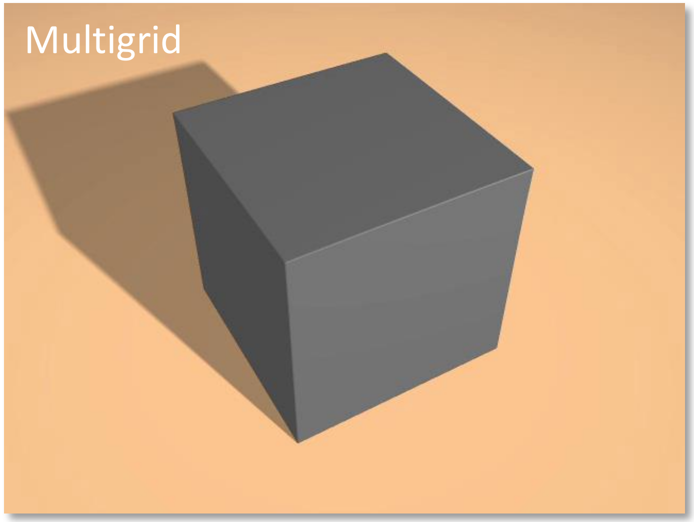
Solver Comparison

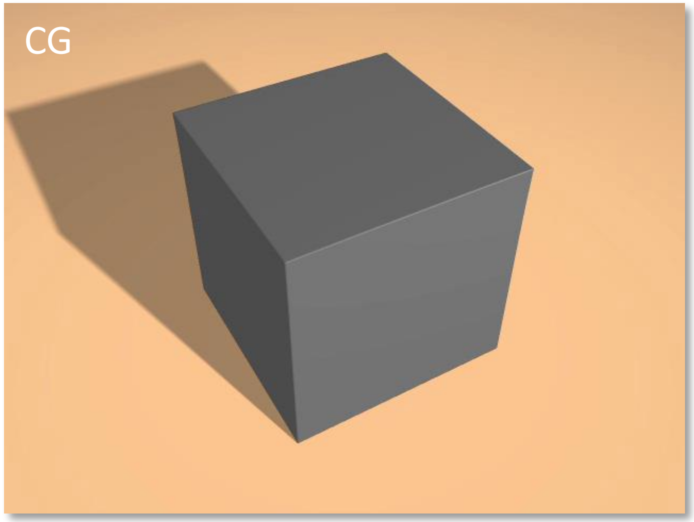
- Comparison wrt run-time (230k elements)



*CPU, Single Core

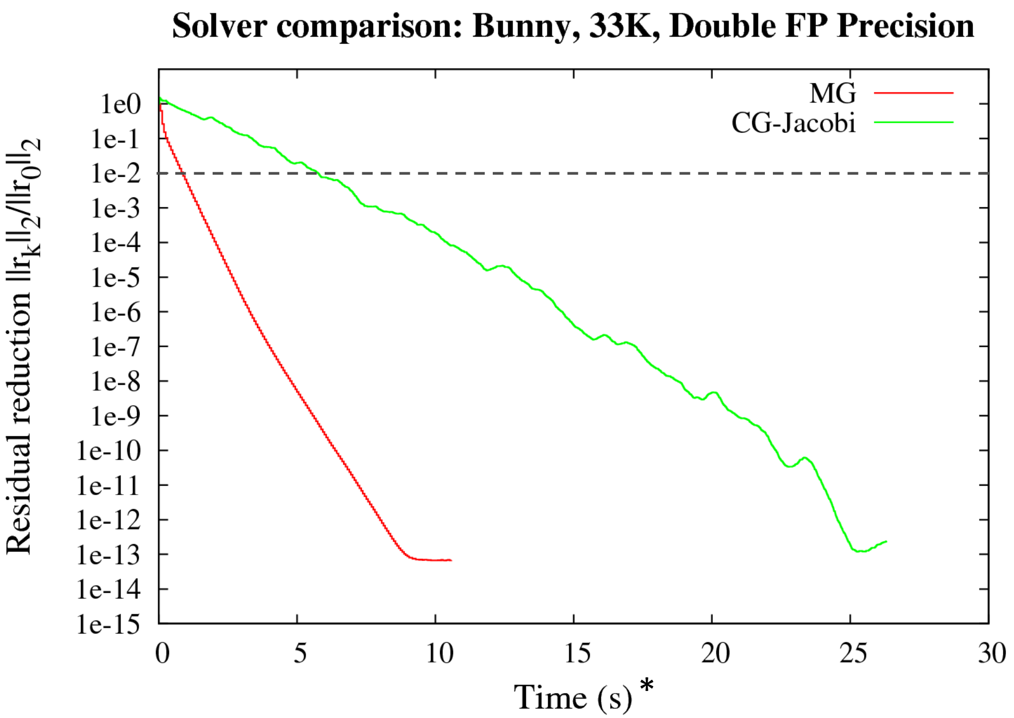
[Dick et al. 2011]



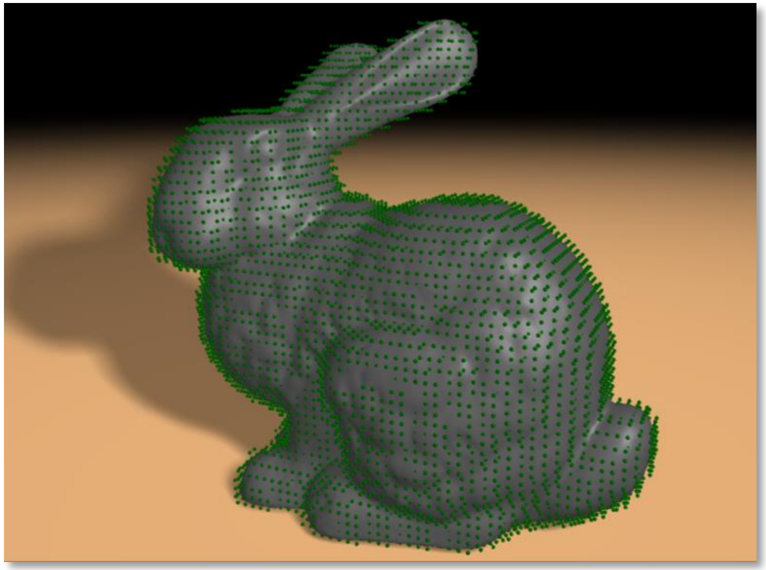


Solver Comparison

- Comparison wrt run-time (33k elements)



*CPU, Single Core



[Dick et al. 2011]

Numerical Solvers

- Discussion
  - Direct vs. iterative solvers
  - Blackbox vs. application-specific solvers
  - Speed vs. implementation effort

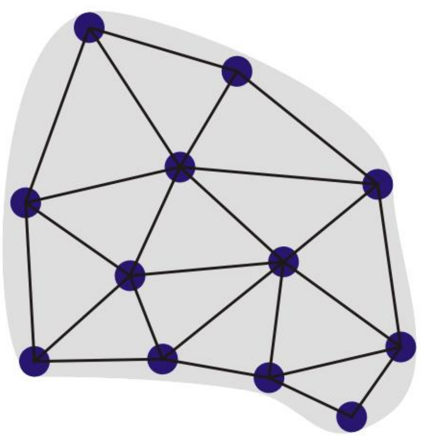
Outline

follows the structure of the report

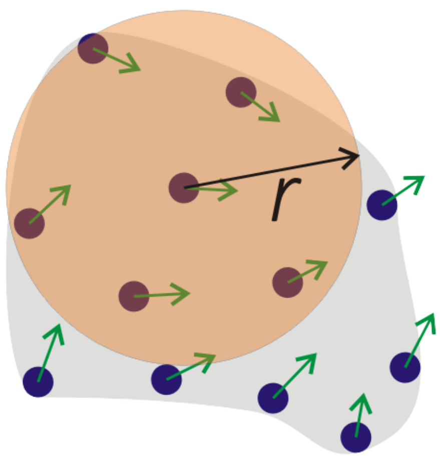
- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation of Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Meshfree Methods

- Model objects as a set of interacting nodes which carry properties, e.g., mass, density, velocity, …
  - Introduced to computer graphics by Desbrun & Cani 1995
  - Re-formulated with continuum mechanics by Müller et al. 2004
- No explicit connectivity information
- Maintain node adjacency implicitly by an influence radius



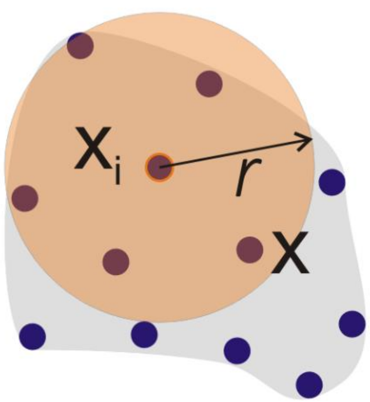
Mesh-based discretization



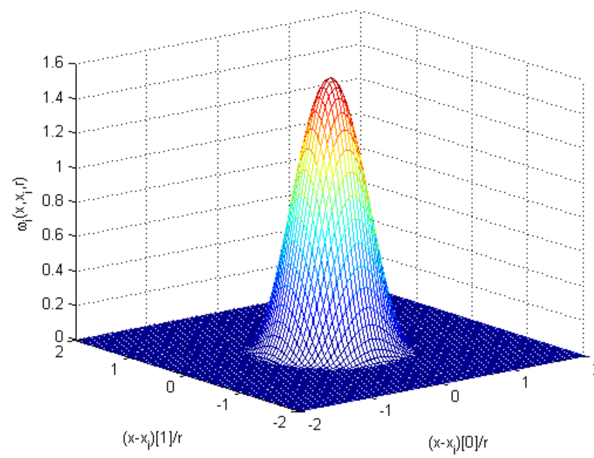
Meshfree discretization

Influence Radius & Weighting Kernel

- Moving Least Squares Approximation [Lancaster & Salkauskas 1981]
- Interpolation: $u(x) = \sum_i \phi_i(x)\, u_i$, for all $i \in \{i \mid d(x, x_i) \leq r\}$
  - r: Influence radius
- Shape function: $\phi_i(x) = \omega_i(x, x_i, r) p^T(x)[M(x)]^{-1} p(x_i)$
  - Polynomial basis of order n: $p(x) = [x^0\ x^1 \ldots x^n]^T$
  - Moment matrix: $M(x) = \sum_i \omega_i(x, x_i, r_i) p(x_i) p^T(x_i)$



Influence radius: r

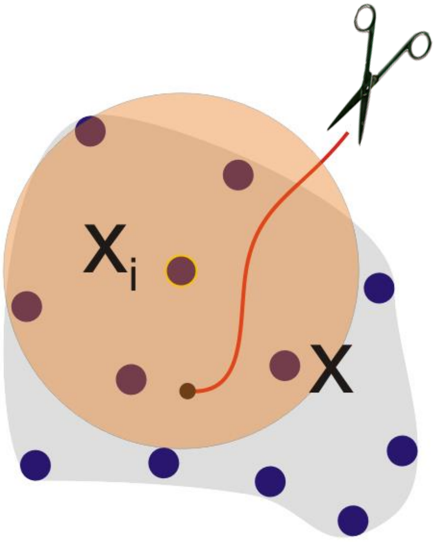


Weighting kernel: $\omega_i(x, x_i, r)$

$$\omega_i(x, x_i, r) = \begin{cases} \frac{315}{64\pi r^3}(r^2 - d^2(x, x_i))^3 & d(x, x_i) \leq r \\ 0 & d(x, x_i) > r \end{cases}$$

Modeling Discontinuity

- Weighting kernel: $\omega_i(x, x_i, r) = \begin{cases} \text{nonzero} & d(x, x_i) \leq r \\ 0 & d(x, x_i) > r \end{cases}$
  - Imply x_i and x are (implicitly) connected if the distance is smaller than the influence radius
- Modeling discontinuity by modifying the weighting kernel

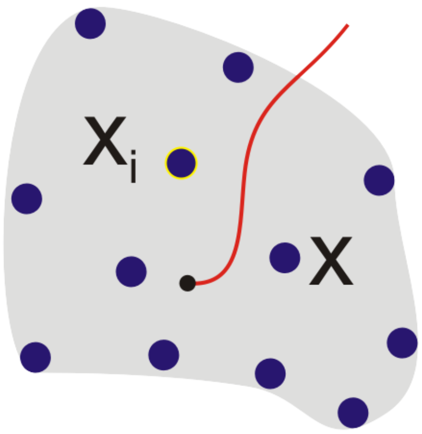


Cutting a meshfree object

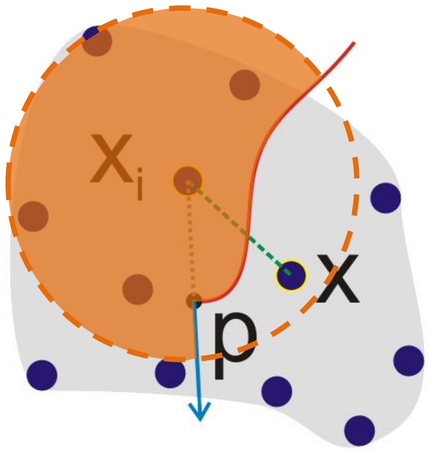
Modeling Discontinuity

- Visibility criterion: assign zero to $\omega_i(x, x_i, r)$, if x is invisible from x_i, i.e., $\overrightarrow{xx_i}$ intersects the cutting path [Belytschko et al. 1994]
- Weighting kernel:

$$\omega_i(x, x_i, r) = \begin{cases} \text{nonzero} & d(x, x_i) \leq r \;\wedge\; x \text{ is visible} \\ 0 & d(x, x_i) > r \;\vee\; x \text{ is invisible} \end{cases}$$



Cutting a meshfree object



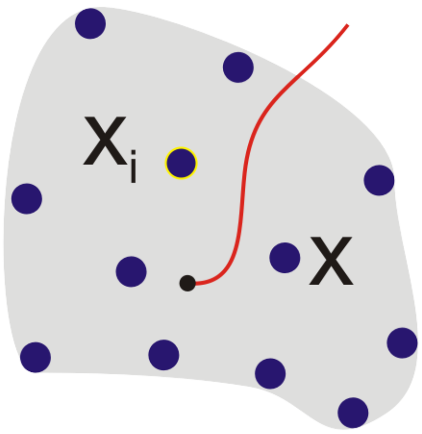
Visibility criterion

Modeling Discontinuity

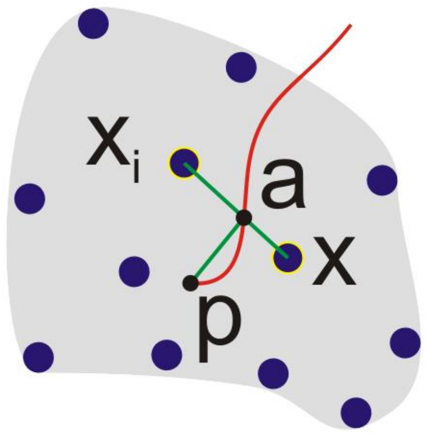
- Transparency method: add to the Euclidean distance $d(x, x_i)$ a factor that depends on the distance $d(p, a)$ [Organ et al. 1996]

- E.g., $\omega_i(x, x_i, r) = \begin{cases} \frac{315}{64\pi r^3}\left(r^2 - d^2(x, x_i)\right)^3 & d(x, x_i) \leq r \\ 0 & d(x, x_i) > r \end{cases}$

$\left(d(x, x_i) + d(p, a)\right)^2$



Cutting a meshfree object



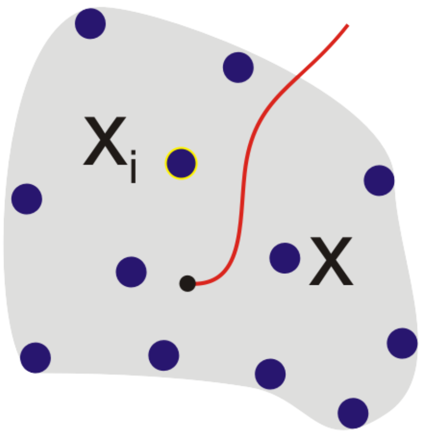
Transparency method

p: the discontinuity tip
a: the intersection

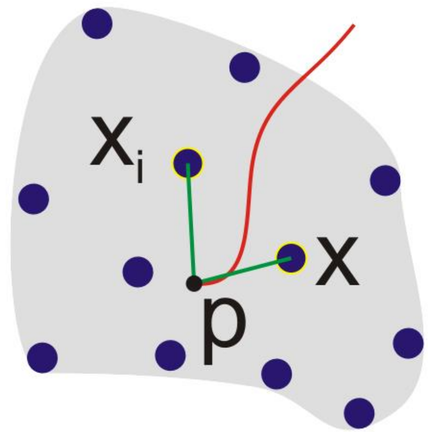
Modeling Discontinuity

- Diffraction method: replace Euclidean distance $d(x, x_i)$ with the distance $d(p, x)$ and $d(p, x_i)$ [Organ et al. 1996]
- E.g., $\omega_i(x, x_i, r) = \begin{cases} \frac{315}{64\pi r^3}(r^2 - d^2(x, x_i))^3 & d(x, x_i) \leq r \\ 0 & d(x, x_i) > r \end{cases}$

$\left(d(p, x) + d(p, x_i)\right)^2$



Cutting a meshfree object



Diffraction method

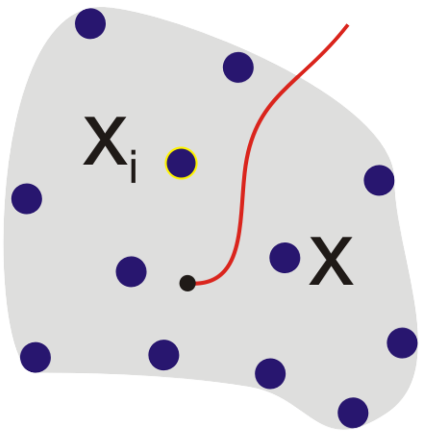
p: the discontinuity tip

In 3D the position of p is not well defined

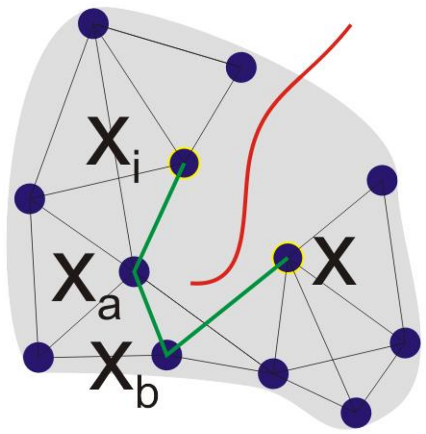
Modeling Discontinuity

- Graph-based diffraction method: replace Euclidean distance $d(x, x_i)$ with the minimum distance $x_i \rightarrow x$ in a graph [Steinemann et al. 2006]
- E.g., $\omega_i(x, x_i, r) = \begin{cases} \frac{315}{64\pi r^3}(r^2 - d^2(x, x_i))^3 & d(x, x_i) \leq r \\ 0 & d(x, x_i) > r \end{cases}$

$d^2(x_i \rightarrow x_a \rightarrow x_b \rightarrow x)$



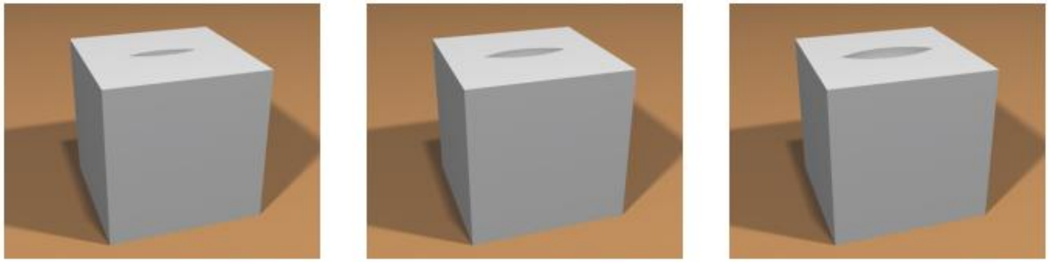
Cutting a meshfree object

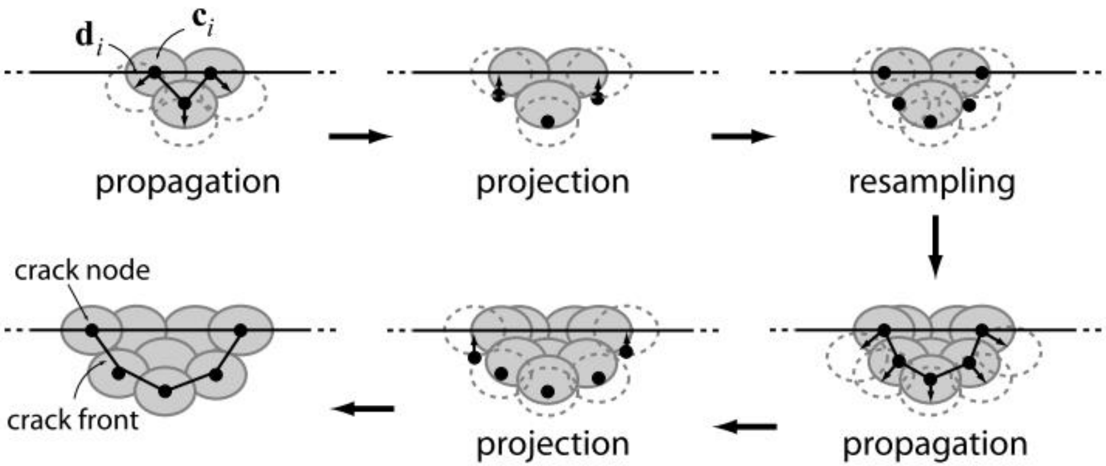


Graph-based diffraction method

Generating New Surface due to Cuts

- Crack surface propagation [Pauly et al. 2005]
  - Represent surface by means of elliptical splats (surfels)
  - Propagate crack front and create additional surfels when necessary



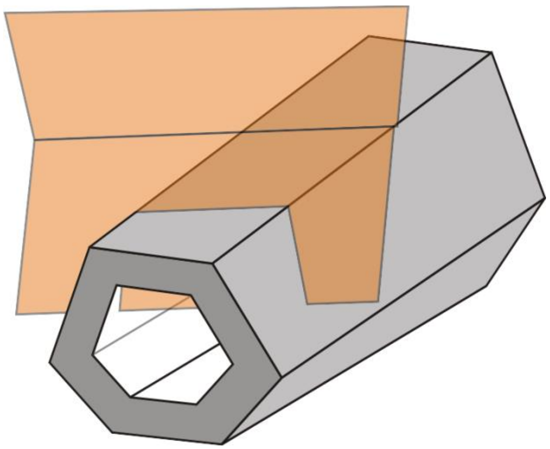


Meshless fracture animation

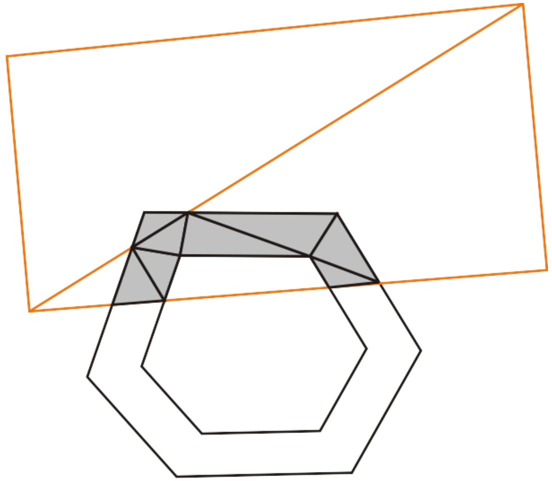
[Pauly et al. 2005]

Generating New Surface due to Cuts

- Explicit cutting surface modeling [Steinemann et al. 2006]
  - Represent cutting surface as a triangle mesh
  - Trim this surface by the initial, triangulated surface of the object



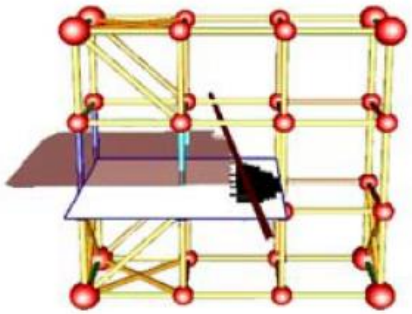
Cutting configuration



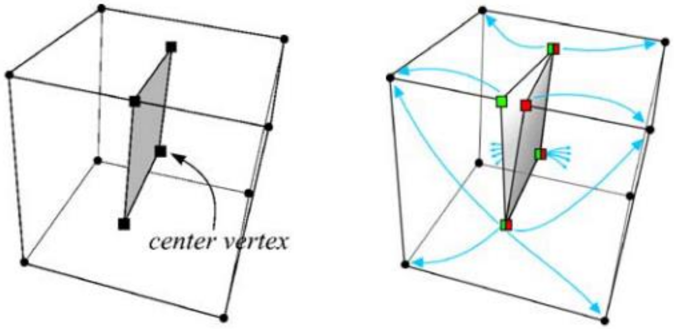
Trimming and triangulation

Generating New Surface due to Cuts

- Surface reconstruction based on a regular hexahedral grid [Pietroni et al. 2009]
  - Deformable body is embedded into a regular hexahedral grid
  - Separate edges of grid cells by cutting tool
  - Reconstruct a triangle mesh from the disconnected edges, using intersection points and normal at these points



Separating of edges



Reconstruction of a triangle mesh

[Pietroni et al 2009]

Discussion on Meshfree Methods

- Advantages:
  - No re-meshing required (volume and surface)
- Disadvantages:
  - Handling of essential boundary conditions is difficult
  - Neighborhood among nodes must be determined during run-time
  - Inversion of the moment matrices is expensive

- Explicit connectivity can still be advantageous …
  - A graph representation can be used to efficiently determine neighborhood [Steinemann et al. 2006]
  - A regular hexahedral grid can be used to contour the surface [Pietroni et al. 2009]

Outline

follows the structure of the report

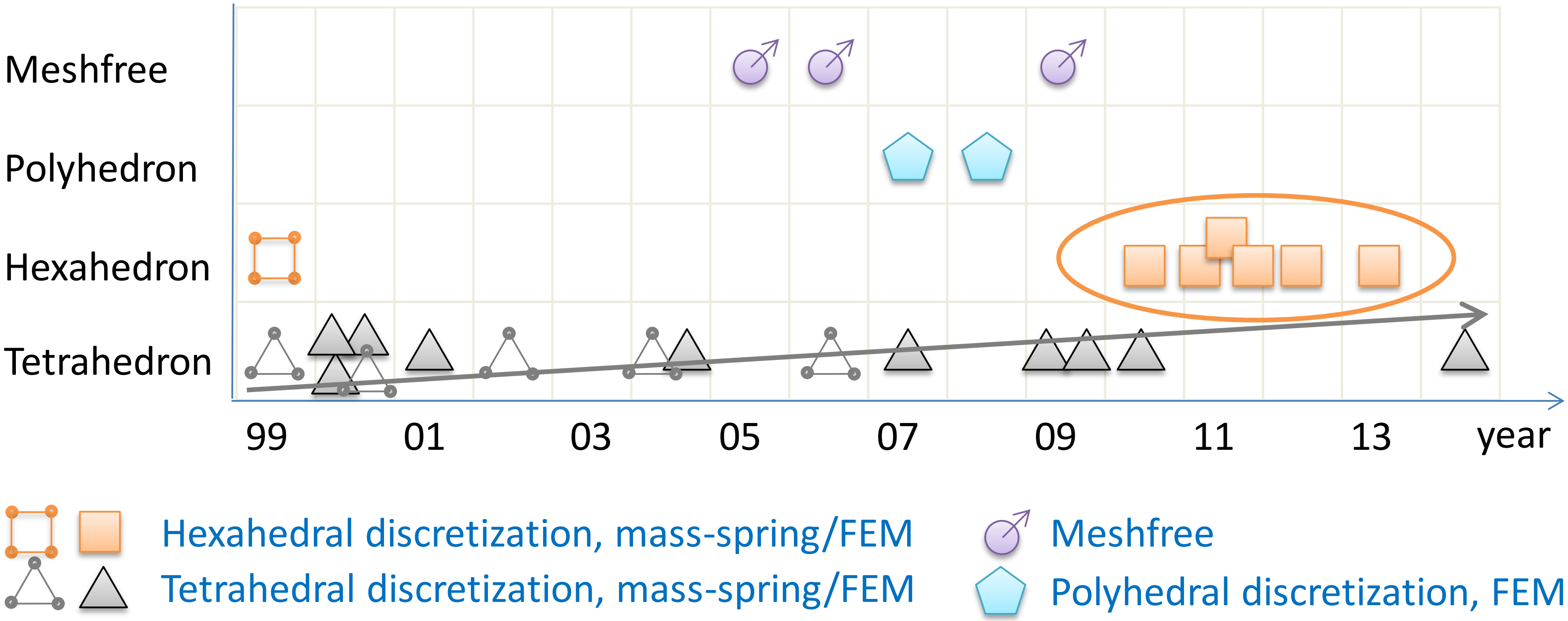
- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation for Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Major Articles Surveyed in this Report

Reference	Geometry	Deformation	Solver	Scenario	Remark
Bielser et al. [BMG99, BG00, BGTG04]	Tet., refinement	Mass-spring	Explicit/Semi-implicit	Interactive	Basic tet. refinement
Cotin et al. [CDA00]	Tet., deletion	Tensor-mass	Explicit	Interactive	Hybrid elastic model
Mor & Kanade [MK00]	Tet., refinement	FEM	Explicit	Interactive	Progressive cutting
Nienhuys et al. [NFvdS00, NFvdS01]	Tet., boundary splitting/snapping	FEM	Static (CG solver)	Interactive	FEM with a CG solver
Bruyns et al. [BSM*02]	Tet., refinement	Mass-spring	Explicit	Interactive	An early survey
Steinemann et al. [SHGS06]	Tet., refinement + snapping	Mass-spring	Explicit	Interactive (Fig. 13 a)	Hybrid cutting
Chentanez et al. [CAR*09]	Tet., refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 d)	Needle insertion
Courtecuisse et al. [CJA*10, CAK*14]	Tet., deletion/refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 c,e)	Surgery applications
Molino et al. [MBF04]	Tet., duplication	FEM	Mixed explicit/implicit	Offline	Basic virtual node algorithm
Sifakis et al. [SDF07]	Tet., duplication	FEM		Offline (Fig. 12 a)	Arbitrary cutting
Jeřábková & Kuhlen [JK09]	Tet.	XFEM	Implicit (CG solver)	Interactive	Introduction of XFEM
Turkiyyah et al. [TKAN09]	Tri.	2D-XFEM	Static (direct solver)	Interactive	XFEM with a direct solver
Kaufmann et al. [KMB*09]	Tri./Quad.	2D-XFEM	Semi-implicit	Offline (Fig. 12 c)	Enrichment textures
Frisken-Gibson [FG99]	Hex., deletion	ChainMail	Local relaxation	Interactive	Linked volume
Jeřábková et al. [JBB*10]	Hex., deletion	CFEM		Interactive	CFEM
Dick et al. [DGW11a]	Hex., refinement	FEM	Implicit (multigrid)	Offline/Interactive (Fig. 12 d)	Linked octree, multigrid solver
Seiler et al. [SSSH11]	Hex., refinement	FEM	Implicit	Interactive	Octree, surface embedding
Wu et al. [WDW11, WBWD12, WDW13]	Hex., refinement	CFEM	Implicit (multigrid)	Interactive (Fig. 13 b, f)	Collision detection for CFEM
Wicke et al. [WBG07]	Poly., splitting	PFEM	Implicit	Offline (Fig. 12 b)	Basic polyhedral FEM
Martin et al. [MKB*08]	Poly., splitting	PFEM	Semi-implicit	Offline	Harmonic basis functions
Pauly et al. [PKA*05]	Particles, transparency	Meshfree	Explicit	Offline	Fracture animation
Steinemann et al. [SOG06]	Particles, diffraction	Meshfree		Offline/Interactive (Fig. 12 e)	Splitting fronts propagation
Pietroni et al. [PGCS09]	Particles, visibility	Meshfree		Interactive	Splitting cubes algorithm

Publication Year – Method Plot

- Trends: from mass-spring systems to finite element methods
- Tetrahedral elements are consistently improved
- Hexahedral elements are recently advocated

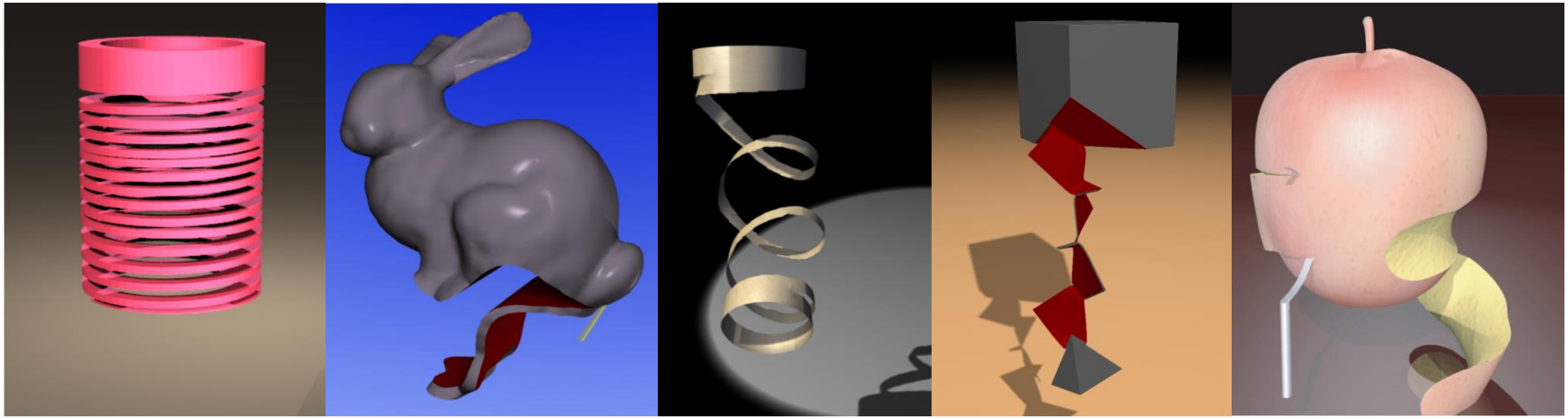


Overview

- Geometrically accurate separation can be supported by all spatial discretizations

Polyhedral FEM
[Wicke et al 2007]

Hexahedral FEM
[Dick et al 2011]



Tetrahedral,
virtual node algorithm
[Sifakis et al 2007]

Quadrilateral,
extended FEM
[Kaufmann et al. 2009]

Meshfree
[Steinemann et al 2011]

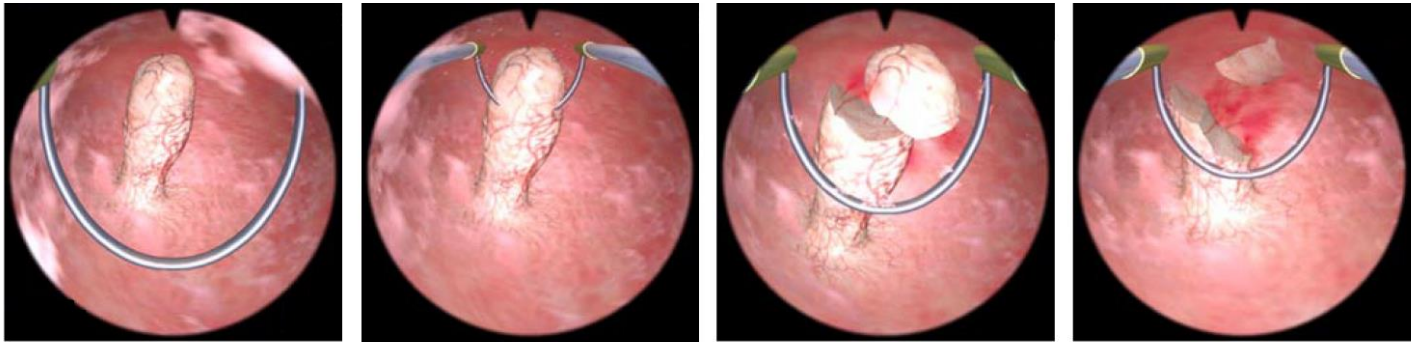
Overview

- Tetrahedral discretizations are widely employed in virtual cutting in surgery simulators

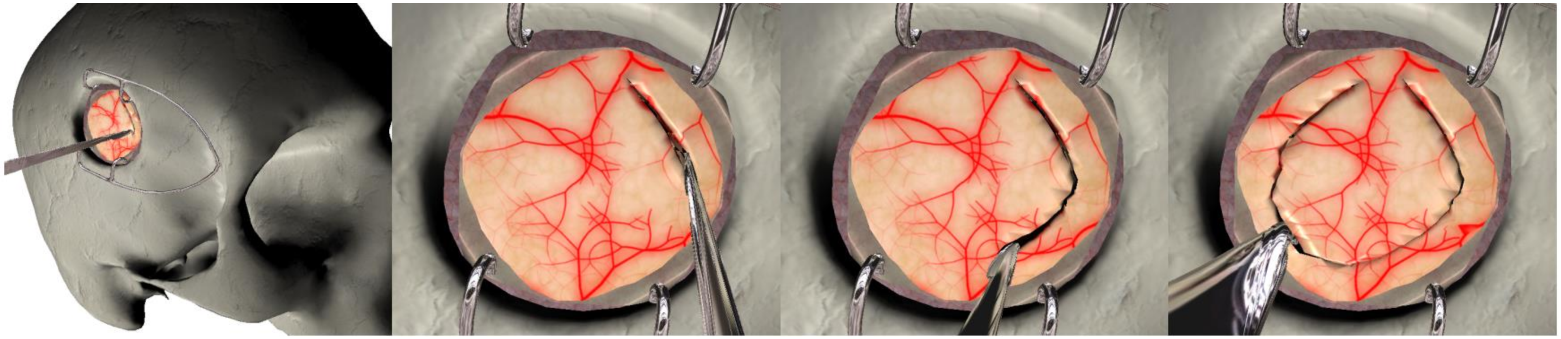
Reference	Geometry	Deformation	Solver	Scenario	Remark
Bielser et al. [BMG99, BG00, BGTG04]	Tet., refinement	Mass-spring	Explicit/Semi-implicit	Interactive	Basic tet. refinement
Cotin et al. [CDA00]	Tet., deletion	Tensor-mass	Explicit	Interactive	Hybrid elastic model
Mor & Kanade [MK00]	Tet., refinement	FEM	Explicit	Interactive	Progressive cutting
Nienhuys et al. [NFvdS00, NFvdS01]	Tet., boundary splitting/snapping	FEM	Static (CG solver)	Interactive	FEM with a CG solver
Bruyns et al. [BSM*02]	Tet., refinement	Mass-spring	Explicit	Interactive	An early survey
Steinemann et al. [SHGS06]	Tet., refinement + snapping	Mass-spring	Explicit	Interactive (Fig. 13 a)	Hybrid cutting
Chentanez et al. [CAR*09]	Tet., refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 d)	Needle insertion
Courtecuisse et al. [CJA*10, CAK*14]	Tet., deletion/refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 c,e)	Surgery applications
Molino et al. [MBF04]	Tet., duplication	FEM	Mixed explicit/implicit	Offline	Basic virtual node algorithm
Sifakis et al. [SDF07]	Tet., duplication	FEM		Offline (Fig. 12 a)	Arbitrary cutting
Jeřábková & Kuhlen [JK09]	Tet.	XFEM	Implicit (CG solver)	Interactive	Introduction of XFEM
Turkiyyah et al. [TKAN09]	Tri.	2D-XFEM	Static (direct solver)	Interactive	XFEM with a direct solver
Kaufmann et al. [KMB*09]	Tri./Quad.	2D-XFEM	Semi-implicit	Offline (Fig. 12 c)	Enrichment textures
Frisken-Gibson [FG99]	Hex., deletion	ChainMail	Local relaxation	Interactive	Linked volume
Jeřábková et al. [JBB*10]	Hex., deletion	CFEM		Interactive	CFEM
Dick et al. [DGW11a]	Hex., refinement	FEM	Implicit (multigrid)	Offline/Interactive (Fig. 12 d)	Linked octree, multigrid solver
Seiler et al. [SSSH11]	Hex., refinement	FEM	Implicit	Interactive	Octree, surface embedding
Wu et al. [WDW11, WBWD12, WDW13]	Hex., refinement	CFEM	Implicit (multigrid)	Interactive (Fig. 13 b, f)	Collision detection for CFEM
Wicke et al. [WBG07]	Poly., splitting	PFEM	Implicit	Offline (Fig. 12 b)	Basic polyhedral FEM
Martin et al. [MKB*08]	Poly., splitting	PFEM	Semi-implicit	Offline	Harmonic basis functions
Pauly et al. [PKA*05]	Particles, transparency	Meshfree	Explicit	Offline	Fracture animation
Steinemann et al. [SOG06]	Particles, diffraction	Meshfree		Offline/Interactive (Fig. 12 e)	Splitting fronts propagation
Pietroni et al. [PGCS09]	Particles, visibility	Meshfree		Interactive	Splitting cubes algorithm

Overview

- Tetrahedral discretizations are widely employed in virtual cutting in surgery simulators



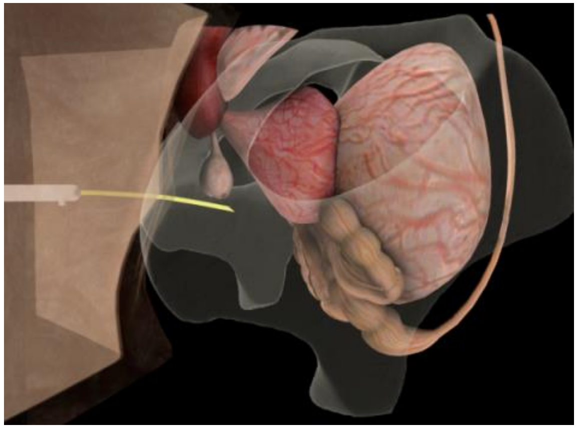
Ablating a polyp in a hysteroscopy simulator [Steinemann et al 2006]



Simulation of a brain tumor resection [Courtecuisse et al 2014]

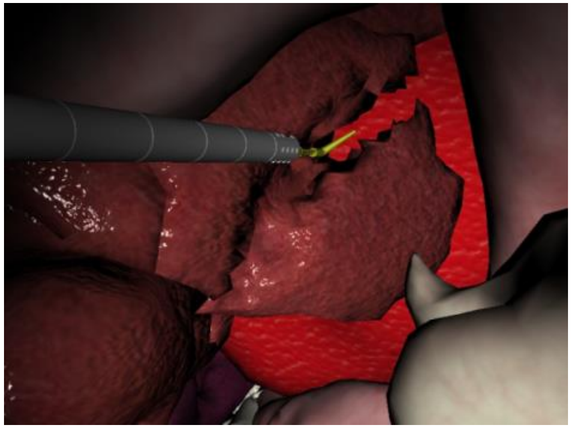
Overview

- Tetrahedral discretizations are widely employed in virtual cutting in surgery simulators



Needle insertion in a prostate brachytherapy simulator

[Chentanez et al 2009]



Real-time simulation of laparoscopic hepatectomy

[Courtecuisse et al 2010]

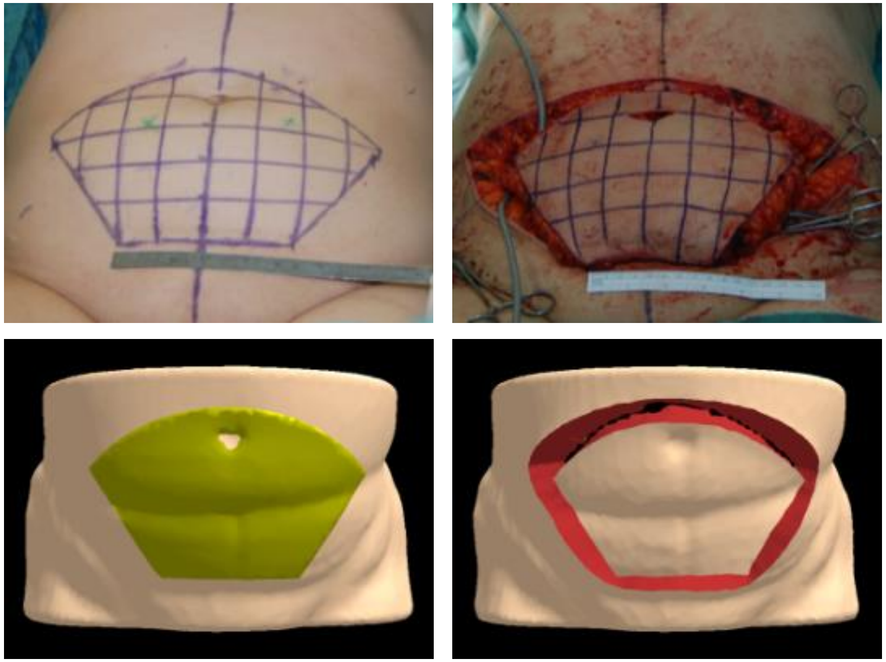
Overview

- Hexahedral discretizations are recently demonstrated to provide a good balance between speed and accuracy

Reference	Geometry	Deformation	Solver	Scenario	Remark
Bielser et al. [BMG99, BG00, BGTG04]	Tet., refinement	Mass-spring	Explicit/Semi-implicit	Interactive	Basic tet. refinement
Cotin et al. [CDA00]	Tet., deletion	Tensor-mass	Explicit	Interactive	Hybrid elastic model
Mor & Kanade [MK00]	Tet., refinement	FEM	Explicit	Interactive	Progressive cutting
Nienhuys et al. [NFvdS00, NFvdS01]	Tet., boundary splitting/snapping	FEM	Static (CG solver)	Interactive	FEM with a CG solver
Bruyns et al. [BSM*02]	Tet., refinement	Mass-spring	Explicit	Interactive	An early survey
Steinemann et al. [SHGS06]	Tet., refinement + snapping	Mass-spring	Explicit	Interactive (Fig. 13 a)	Hybrid cutting
Chentanez et al. [CAR*09]	Tet., refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 d)	Needle insertion
Courtecuisse et al. [CJA*10, CAK*14]	Tet., deletion/refinement	FEM	Implicit (CG solver)	Interactive (Fig. 13 c,e)	Surgery applications
Molino et al. [MBF04]	Tet., duplication	FEM	Mixed explicit/implicit	Offline	Basic virtual node algorithm
Sifakis et al. [SDF07]	Tet., duplication	FEM		Offline (Fig. 12 a)	Arbitrary cutting
Jeřábková & Kuhlen [JK09]	Tet.	XFEM	Implicit (CG solver)	Interactive	Introduction of XFEM
Turkiyyah et al. [TKAN09]	Tri.	2D-XFEM	Static (direct solver)	Interactive	XFEM with a direct solver
Kaufmann et al. [KMB*09]	Tri./Quad.	2D-XFEM	Semi-implicit	Offline (Fig. 12 c)	Enrichment textures
Frisken-Gibson [FG99]	Hex., deletion	ChainMail	Local relaxation	Interactive	Linked volume
Jeřábková et al. [JBB*10]	Hex., deletion	CFEM		Interactive	CFEM
Dick et al. [DGW11a]	Hex., refinement	FEM	Implicit (multigrid)	Offline/Interactive (Fig. 12 d)	Linked octree, multigrid solver
Seiler et al. [SSSH11]	Hex., refinement	FEM	Implicit	Interactive	Octree, surface embedding
Wu et al. [WDW11, WBWD12, WDW13]	Hex., refinement	CFEM	Implicit (multigrid)	Interactive (Fig. 13 b, f)	Collision detection for CFEM
Wicke et al. [WBG07]	Poly., splitting	PFEM	Implicit	Offline (Fig. 12 b)	Basic polyhedral FEM
Martin et al. [MKB*08]	Poly., splitting	PFEM	Semi-implicit	Offline	Harmonic basis functions
Pauly et al. [PKA*05]	Particles, transparency	Meshfree	Explicit	Offline	Fracture animation
Steinemann et al. [SOG06]	Particles, diffraction	Meshfree		Offline/Interactive (Fig. 12 e)	Splitting fronts propagation
Pietroni et al. [PGCS09]	Particles, visibility	Meshfree		Interactive	Splitting cubes algorithm

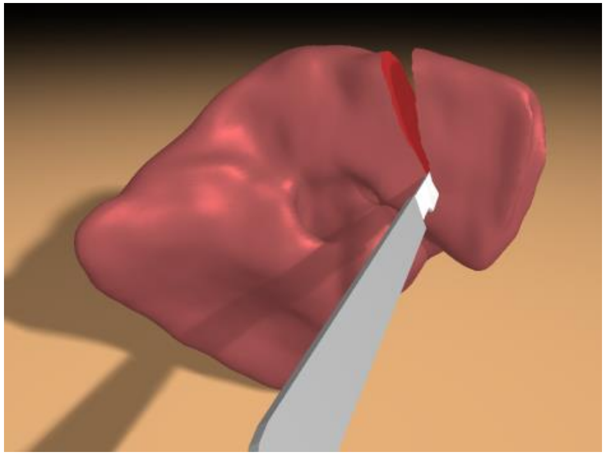
Overview

- Hexahedral discretizations are recently demonstrated to provide a good balance between speed and accuracy



Virtual soft tissue cutting and shrinkage simulation

[Wu et al 2012]



Haptic-enabled virtual cutting of high-resolution soft tissues

[Wu et al 2014]

Purposes of Application Study

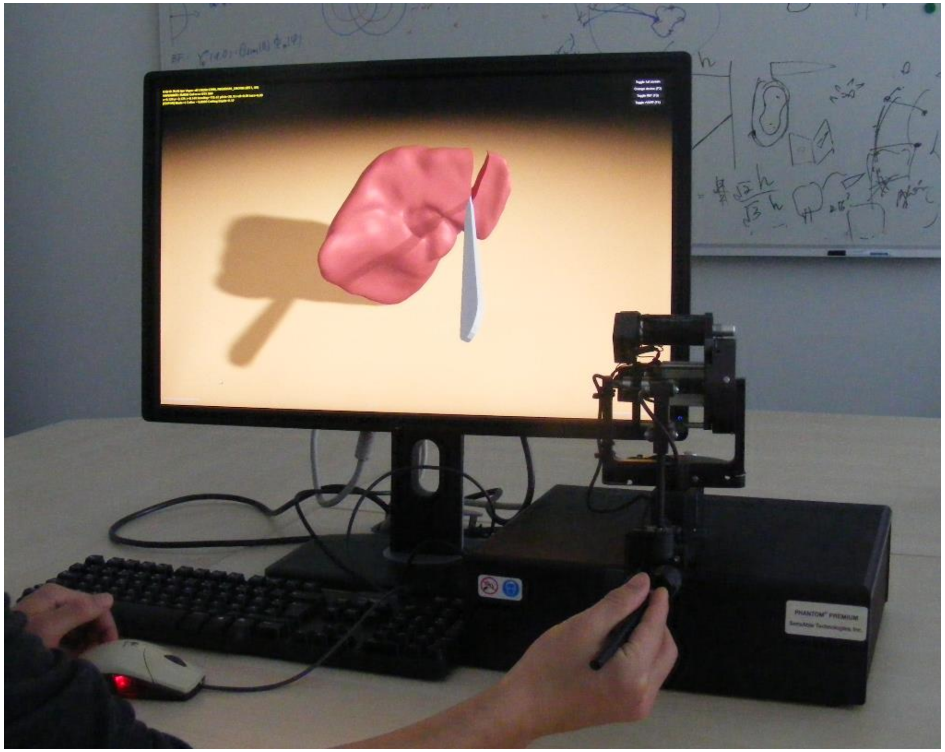
- Provide an estimation of the performance of virtual cutting
- Identify performance bottlenecks in the simulation loop
- Exam accuracy and performance of adaptive methods

- Not an evaluation of all techniques
- But a detailed analysis of our implementations of three variants of hexahedral finite elements

Experimental Setup

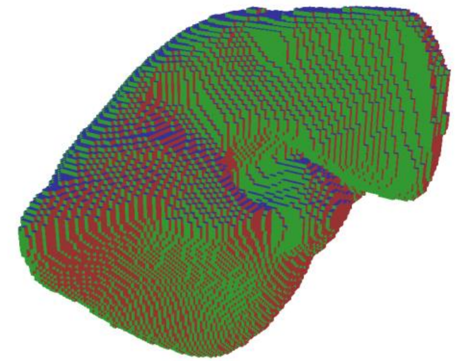
- Linear elastic material, corotational strain formulation

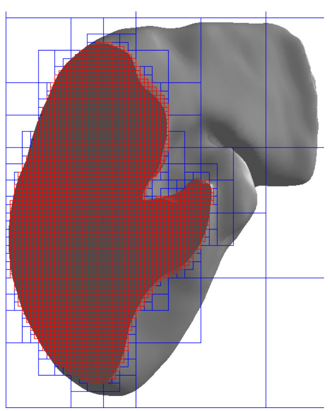
- Standard desktop PC
  - Intel Xeon X5560 processor (a single core was used)
  - 8 GB main memory
- Haptic device
  - Sensable Phantom Premium 1.5

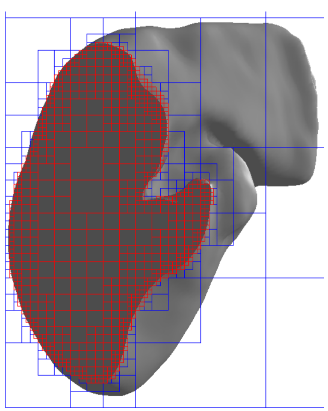


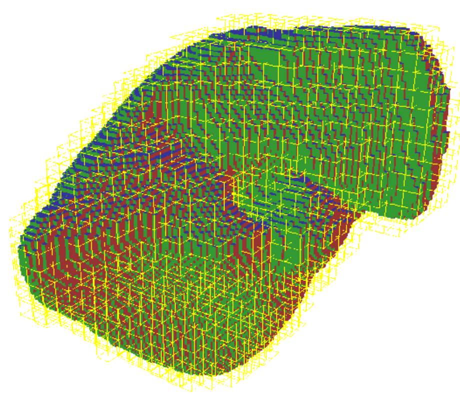
Three Variants

- Basis
  - Geometry modeling: hexahedral elements
  - Surface reconstruction: dual contouring
  - Numerical solver: multigrid solver
- Variants
  - FEs on a uniform hexahedral grid
  - FEs on an adaptive octree grid
  - Composite FEs on an adaptive octree grid

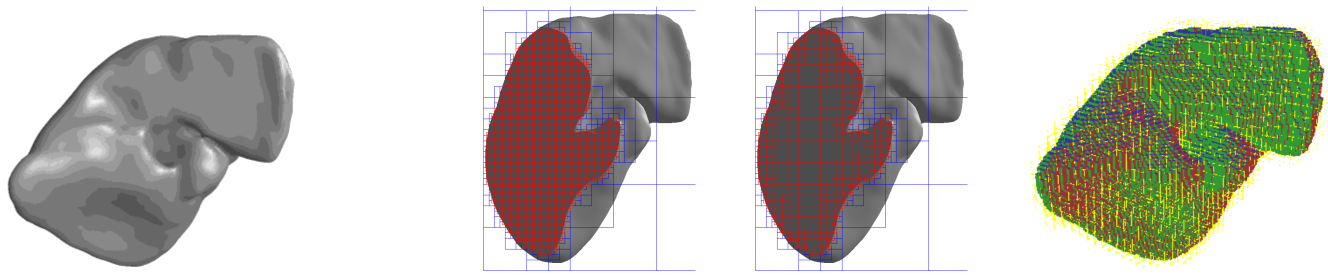








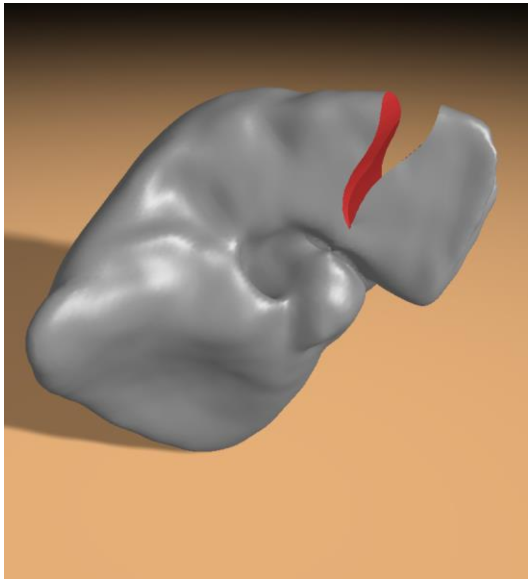
Model Information of Three Variants



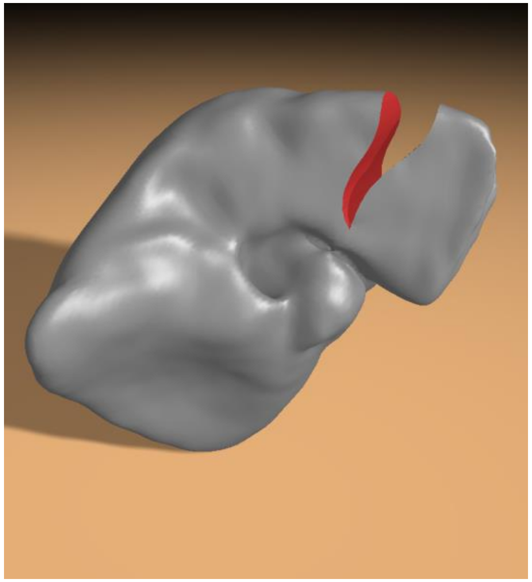
	Uniform	Adaptive	Composite (2 levels)
Coarse resolution		21×21×25	21×21×25
Refined resolution	82×83×100	82×83×100	82×83×100
# Cells (initial)	173 843	40 080	3 439
# DOFs (initial)	566 493	129 162	13 557
# Cells (added due to cut)	0	1 596	39
# DOFs (added due to cut)	2 037	6 438	318

Simulation Results

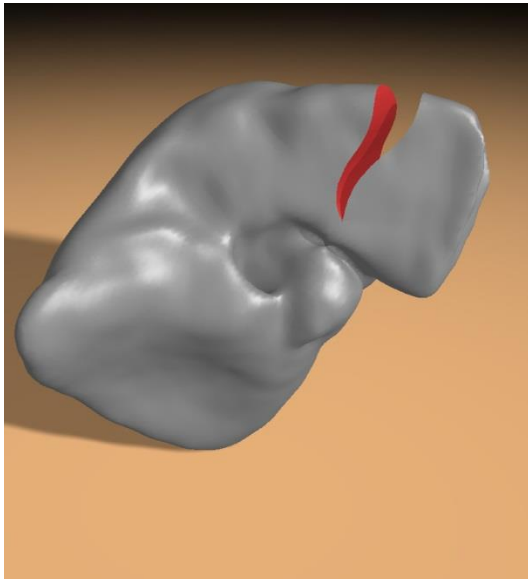
- Adaptive octree deformation resembles the uniform approach
- Composite simulation results in a slightly stiffer deformation



FEs on
a uniform hexahedral grid



FEs on
an adaptive octree grid



Composite FEs on
an adaptive octree grid

Timings

- Accurate cutting simulation can be performed at 2 seconds per frame, on a uniform 82×83×100 grid

	Uniform
Coarse resolution	
Refined resolution	82×83×100
# DOFs (initial)	566 493
Octree subdivision (t_1)	0
Surface meshing (t_2)	1.26
FE matrices (t_3)	29.57
Multigrid hierarchy (t_4)	40.34
Solver (t_5)	2 033.09
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26

Timing
in milliseconds

Timings

- Numerical solver is the bottleneck in cutting simulation

	Uniform
Coarse resolution	
Refined resolution	82×83×100
# DOFs (initial)	566 493
Octree subdivision (t_1)	0
Surface meshing (t_2)	1.26
FE matrices (t_3)	29.57
Multigrid hierarchy (t_4)	40.34
Solver (t_5)	2 033.09
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26

Timing
in milliseconds

Timings

- Adaptive octree improves the performance by a factor of 3.5

	Uniform	Adaptive
Coarse resolution		21×21×25
Refined resolution	82×83×100	82×83×100
# DOFs (initial)	566 493	129 162
Octree subdivision (t_1)	0	13.29
Surface meshing (t_2)	1.26	1.26
FE matrices (t_3)	29.57	7.05
Multigrid hierarchy (t_4)	40.34	10.09
Solver (t_5)	2 033.09	581.66
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26	613.35

Timing
in milliseconds

Timings

- Interactive cutting (12 fps) is possible on a 21×21×25 composite simulation grid

	Uniform	Adaptive	Composite (2 levels)
Coarse resolution		21×21×25	21×21×25
Refined resolution	82×83×100	82×83×100	82×83×100
# DOFs (initial)	566 493	129 162	13 557
Octree subdivision (t_1)	0	13.29	13.39
Surface meshing (t_2)	1.26	1.26	1.24
FE matrices (t_3)	29.57	7.05	20.99
Multigrid hierarchy (t_4)	40.34	10.09	2.06
Solver (t_5)	2 033.09	581.66	40.61
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26	613.35	78.29

Timing in milliseconds

Timings

- Solver, FE matrices, octree subdivision affect the performance in the composite approach

	Uniform	Adaptive	Composite (2 levels)
Coarse resolution		21×21×25	21×21×25
Refined resolution	82×83×100	82×83×100	82×83×100
# DOFs (initial)	566 493	129 162	13 557
Octree subdivision (t_1)	0	13.29	13.39
Surface meshing (t_2)	1.26	1.26	1.24
FE matrices (t_3)	29.57	7.05	20.99
Multigrid hierarchy (t_4)	40.34	10.09	2.06
Solver (t_5)	2 033.09	581.66	40.61
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26	613.35	78.29

Timing in milliseconds

Timings

- Time of surface meshing is negligible

	Uniform	Adaptive	Composite (2 levels)
Coarse resolution		21×21×25	21×21×25
Refined resolution	82×83×100	82×83×100	82×83×100
# DOFs (initial)	566 493	129 162	13 557
Octree subdivision (t_1)	0	13.29	13.39
Surface meshing (t_2)	1.26	1.26	1.24
FE matrices (t_3)	29.57	7.05	20.99
Multigrid hierarchy (t_4)	40.34	10.09	2.06
Solver (t_5)	2 033.09	581.66	40.61
Simulation per cut ($\sum_{i=1}^{5} t_i$)	2 104.26	613.35	78.29

Timing
in milliseconds

Outline

follows the structure of the report

- Introduction
- Mesh-based Modeling of Cuts
- Finite Element Simulation for Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study
- Discussion & Conclusion

Summary

- Mesh-based Modeling of Cuts
- Finite Element Simulation for Virtual Cutting
- Numerical Solvers
- Meshfree Methods
- Summary & Application Study

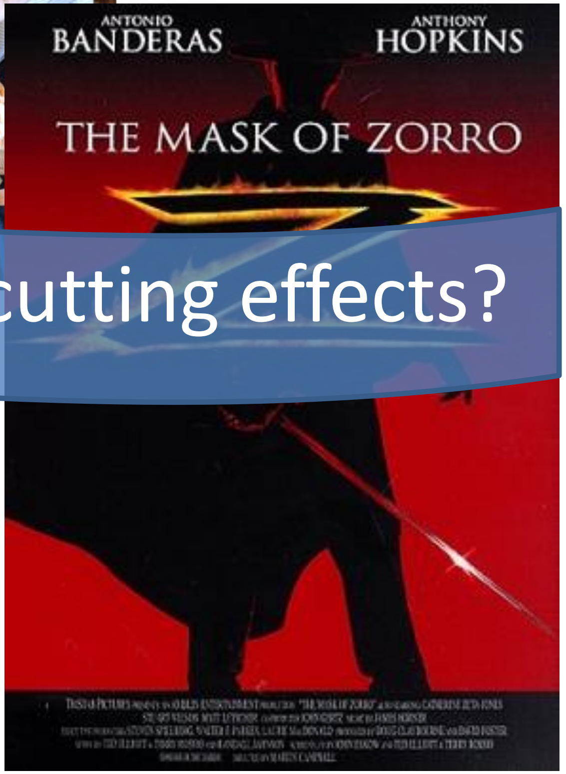
Future Challenges

- Benchmark problems for virtual cutting methods
- Real-world material properties
  - Nonlinear, anisotropic, viscoelastic, viscoplastic materials
- Parallelization on multi-core and multi-GPU architectures
  - Inherently sequential parts
  - Bandwidth and latency bottleneck
- Physical interaction between a scalpel and soft tissues
- Efficient numerical solution techniques on irregular adaptive spatial discretizations

Cutting Is All Around Us!

footage.shutterstock.com

pfollansbee.wordpress.com



How to simulate these interesting cutting effects?

www.hurriyetdailynews.com

wb-3d.com

en.wikipedia.org

Thank you for your attention!

tum.3D

computer graphics & visualization