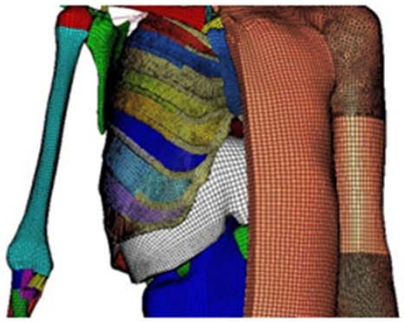
Hexahedral Meshing with Varying Element Sizes

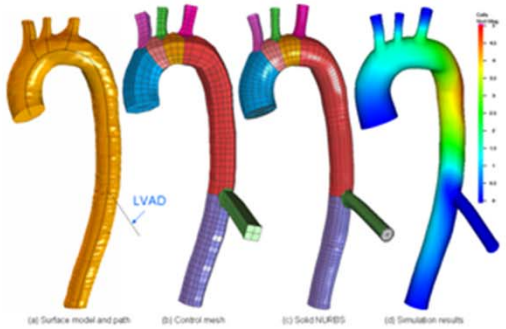
Kaoji Xu, Xifeng Gao, Zhigang Deng, Guoning Chen
University of Houston

Why Hexahedral (Hex-) Meshes?

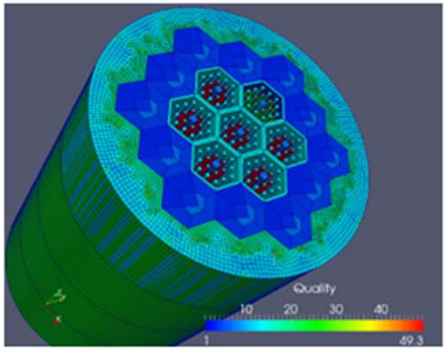
Medical



Biomechanics [Wayne State University]

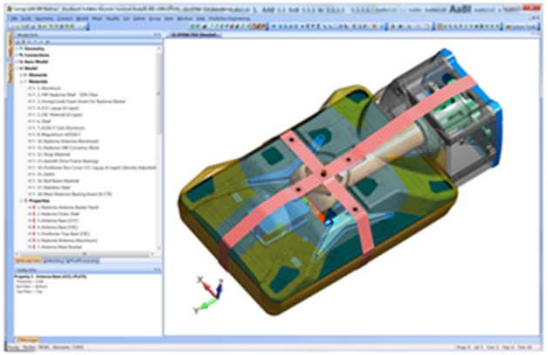


Simulation-based medical planning [Zhang, et al. 2007]



Nuclear reactor manufacturing [Kitware]

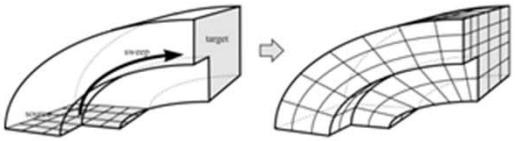
CAD



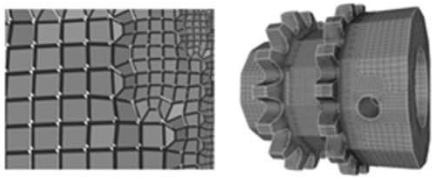
FEA Simulation [Predictive engineering]

Existing Hex-Meshing Techniques

Sweeping/Octree



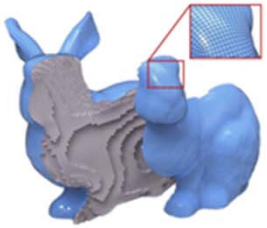
[Owen, et al. 1998]



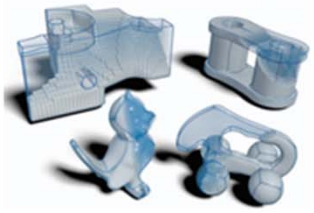
[Maréchal, et al. 2009]

Frame Field

[Nieser, et al. 2011]

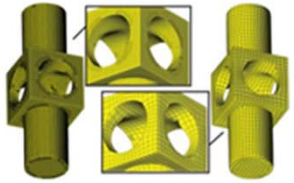


[Li, et al. 2012]



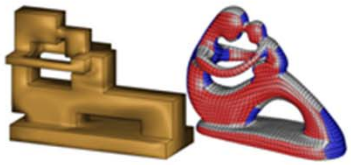
[Jiang, et al. 2013]

Polycube



[Gregson, et al. 2011]

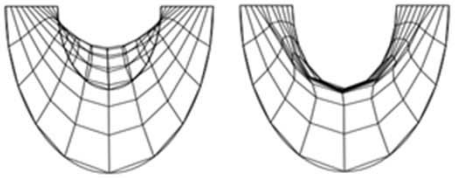
[Huang, et al. 2014]



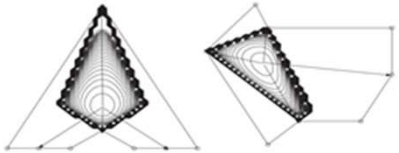
[Livesu, et al. 2013]

Existing Hex-Meshing Optimization Techniques

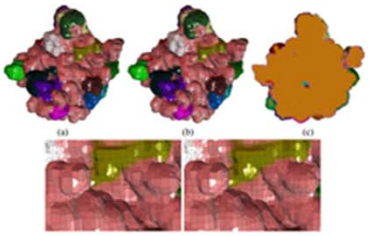
Geometric Optimization



[Knupp, et al. 2000]



[Knupp 2003]

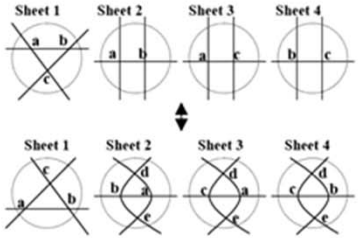


[Zhang, et al. 2009]

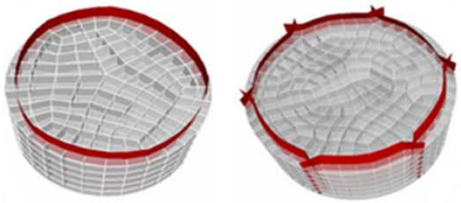
[Ruiz-Gironés, et al. 2014]

[Livesu, et al. 2015]

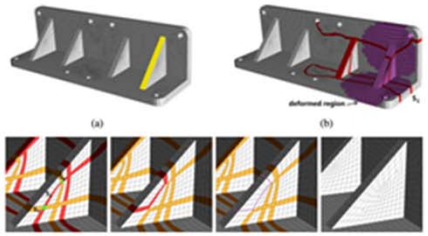
Simplification



[Tautges, et al. 2003]



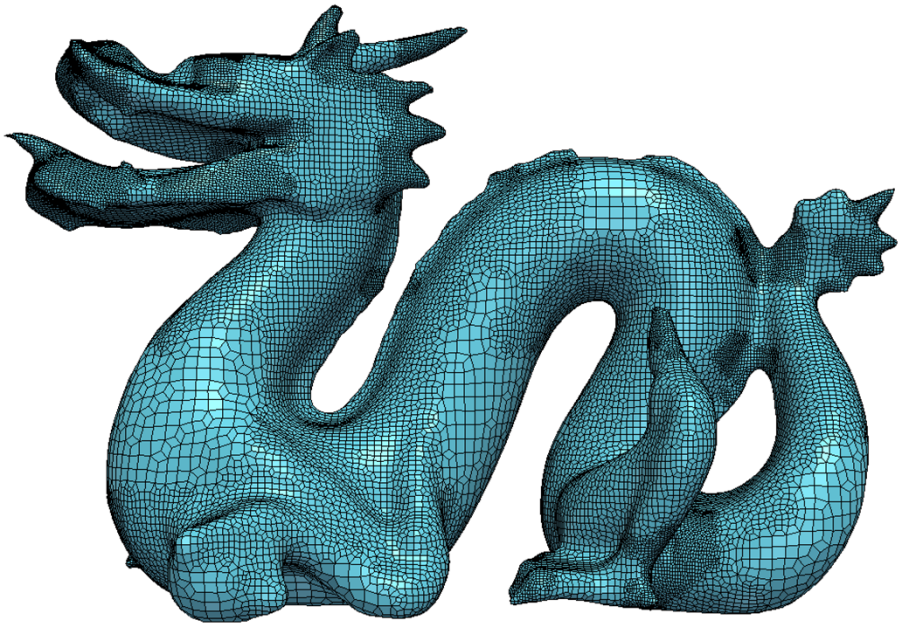
[Ledoux, et al. 2010]



[Zhu, et al. 2014]

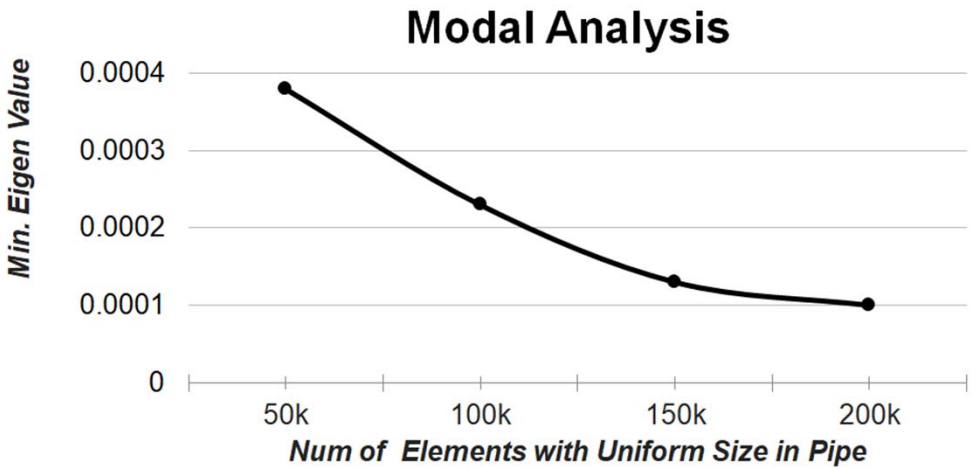
Why Adaptive Meshes?

Adaptive meshes: Element sizes adapt to the feature sizes.

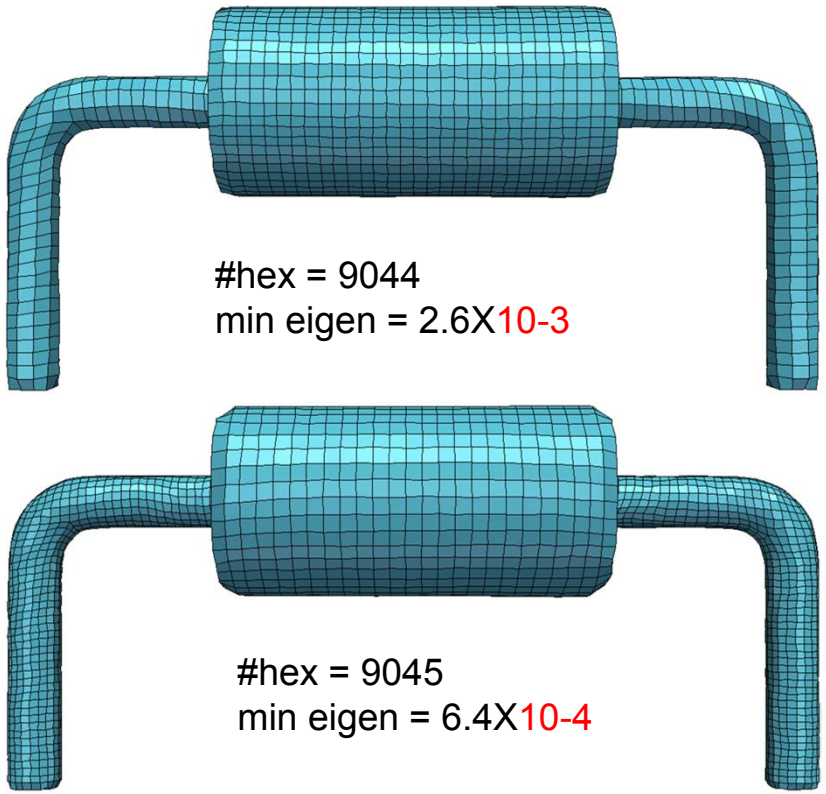


Octree

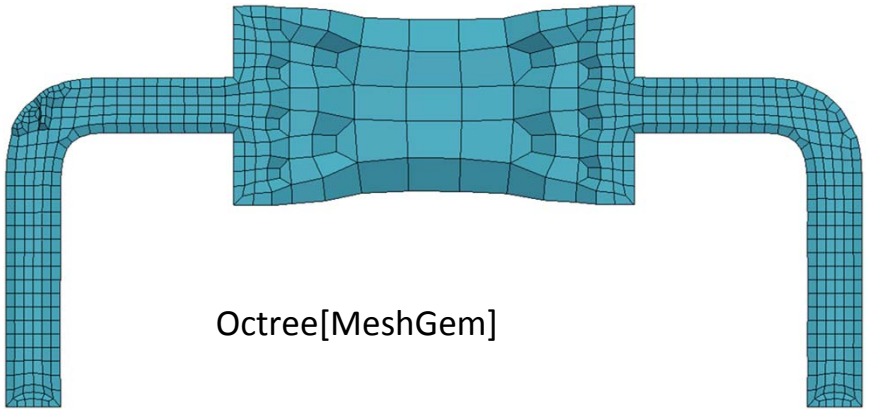
Meshing quality in the linear elasticity problem



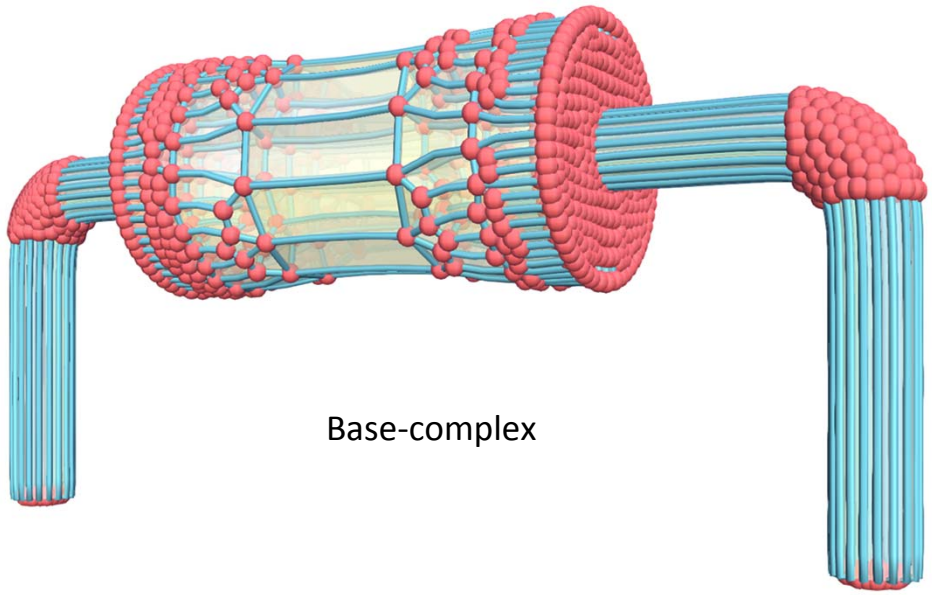
The small the min. eigenvalue, the more accurate in modal analysis



What's the problem? Singularities



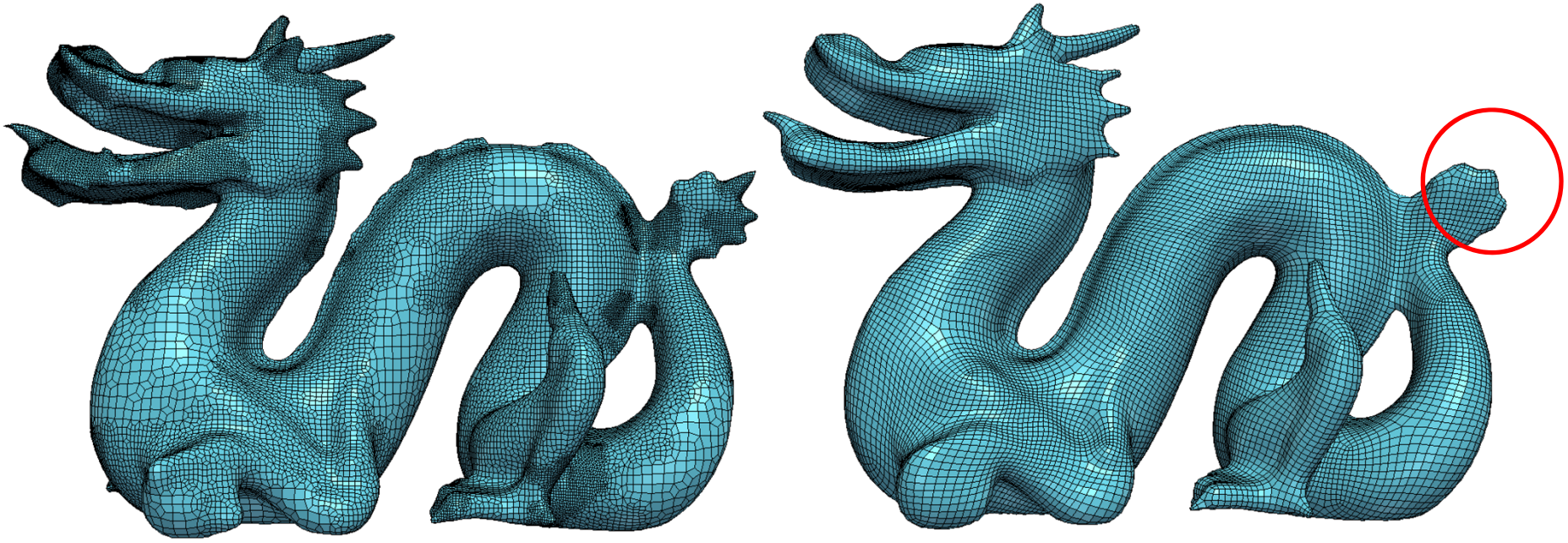
Octree[MeshGem]



Base-complex

What's the problem? Feature Loss

Polycube and frame field based method tend to smooth out surfaces.

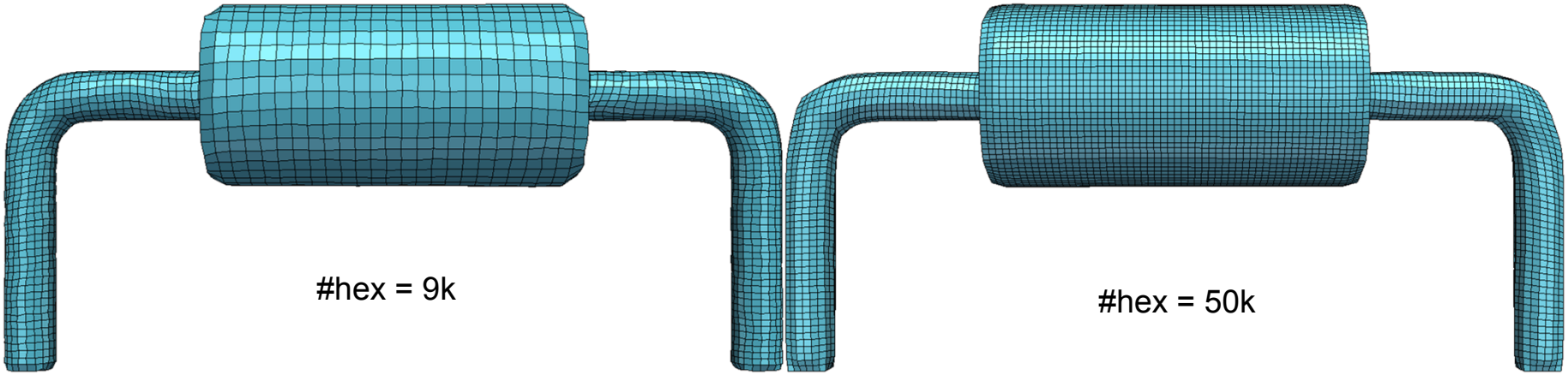


Octree

L1-PolyCube

What's the problem?Too dense!

L1-PolyCube



#hex = 9k

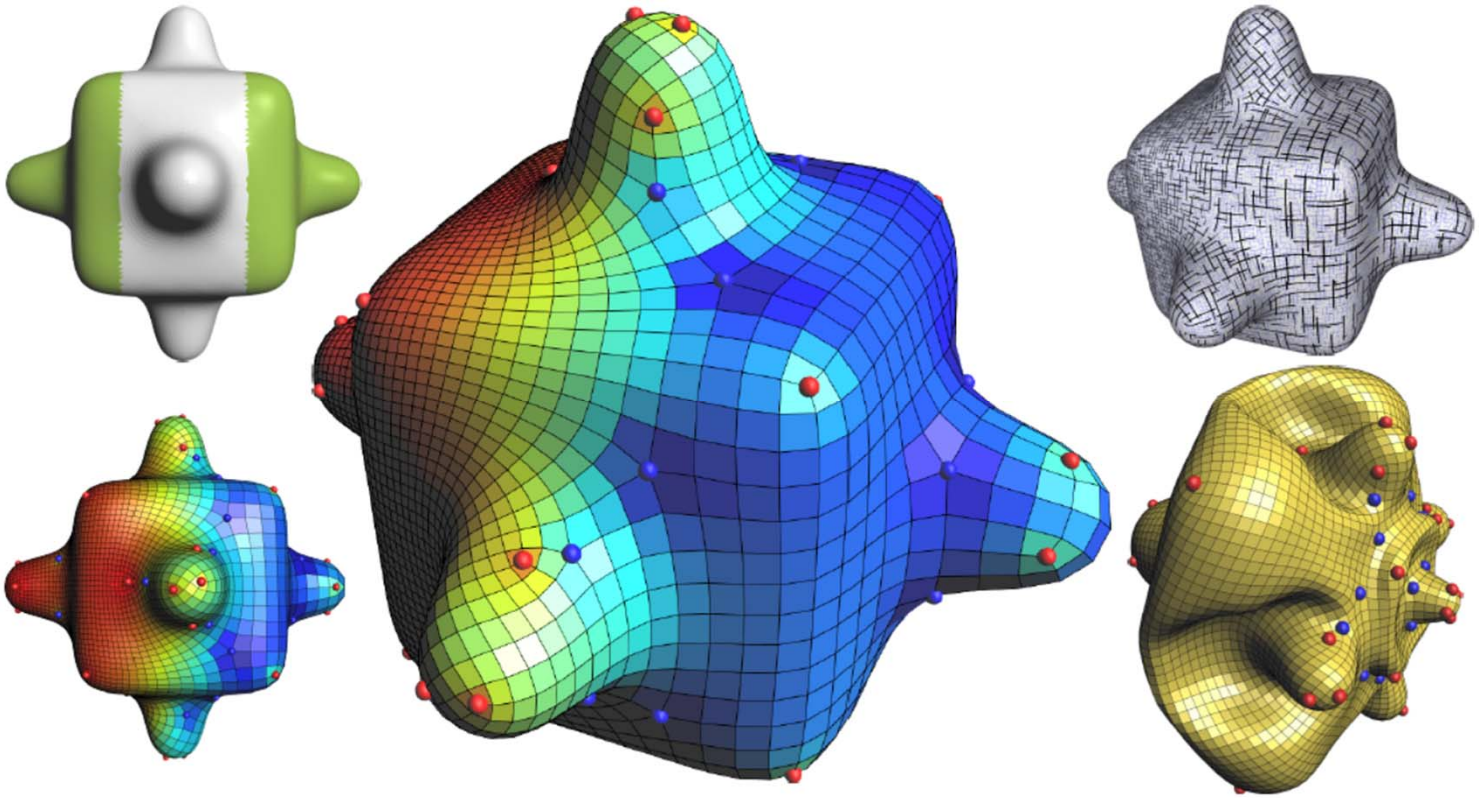
#hex = 50k

preferred

Not preferred

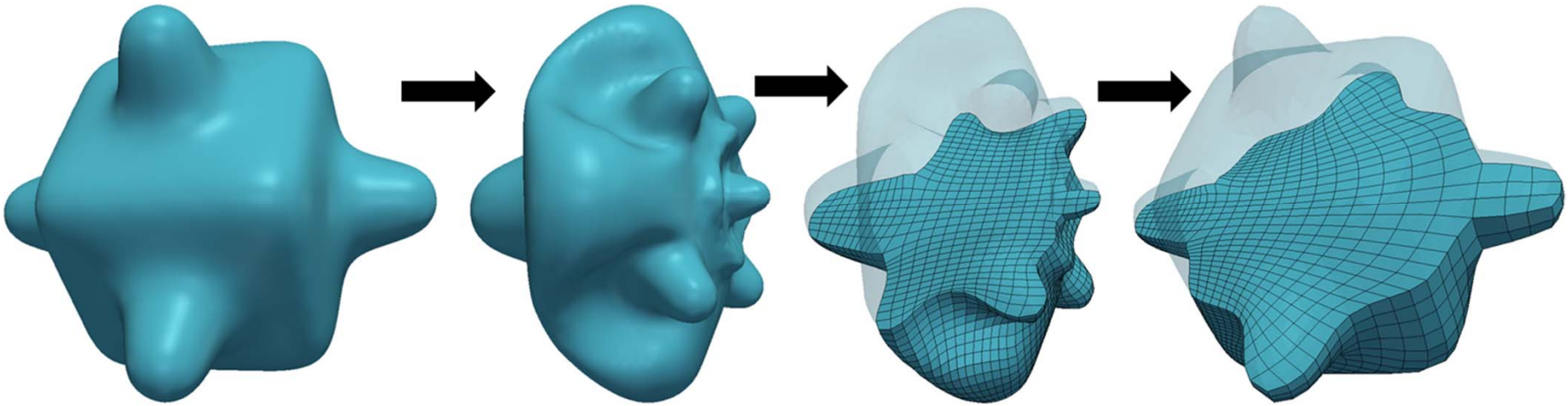
Is there a way to generate adaptive hex-meshes with simple structure?

Inspiration - Anisotropic Quad Meshing



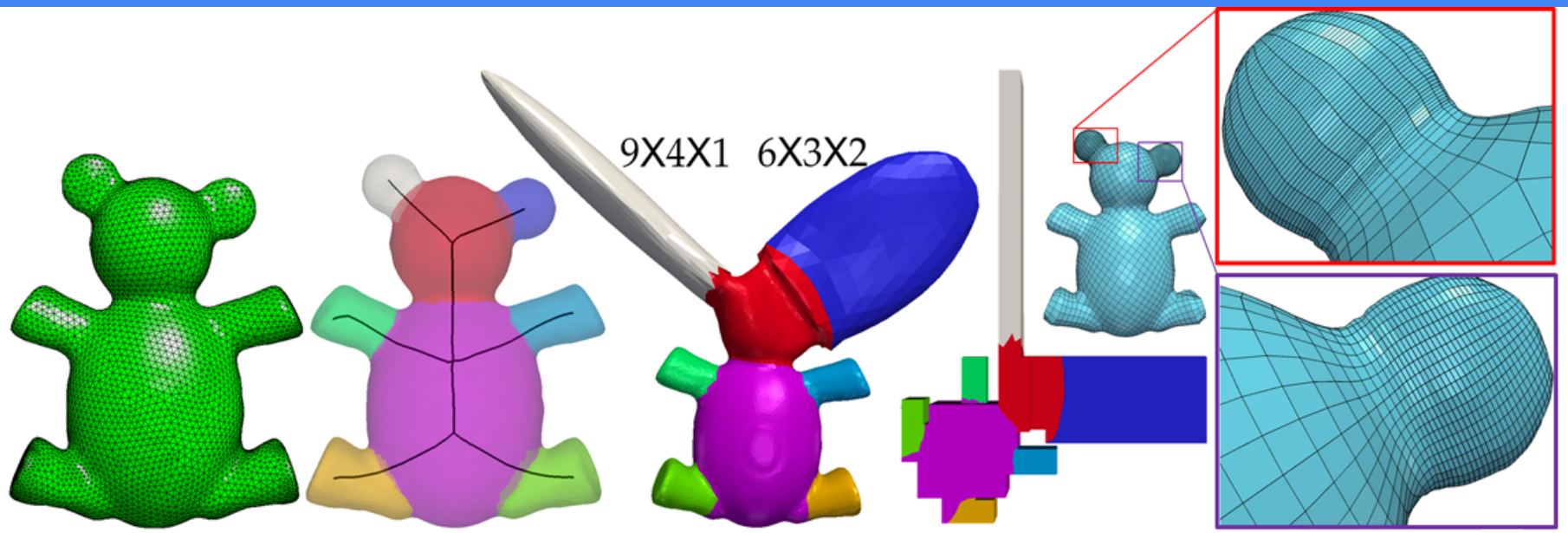
Anisotropic Quad Meshing [Panozzo et al. 2014 - Frame Fields Anisotropic and Non-Orthogonal Cross Fields]

Our Adaption - Anisotropic Hex-Meshing



Deformation Uniform Meshing Mapping

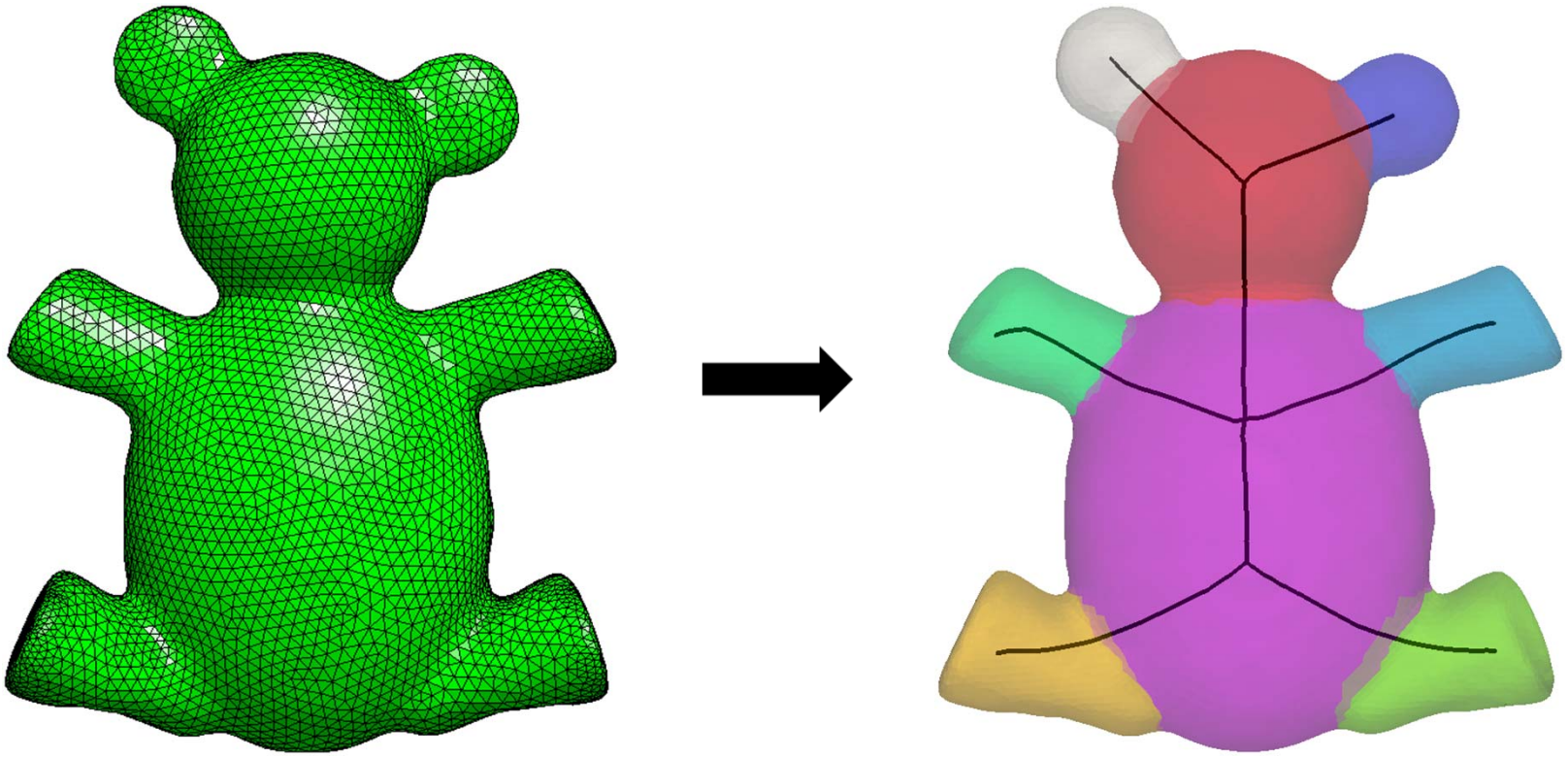
Our Pipeline



Input → Surface segmen-tation & feature size estimation → Magnify ROIs → Generate hex-mesh with uniform element size → Map the hex-mesh back to the original sapce

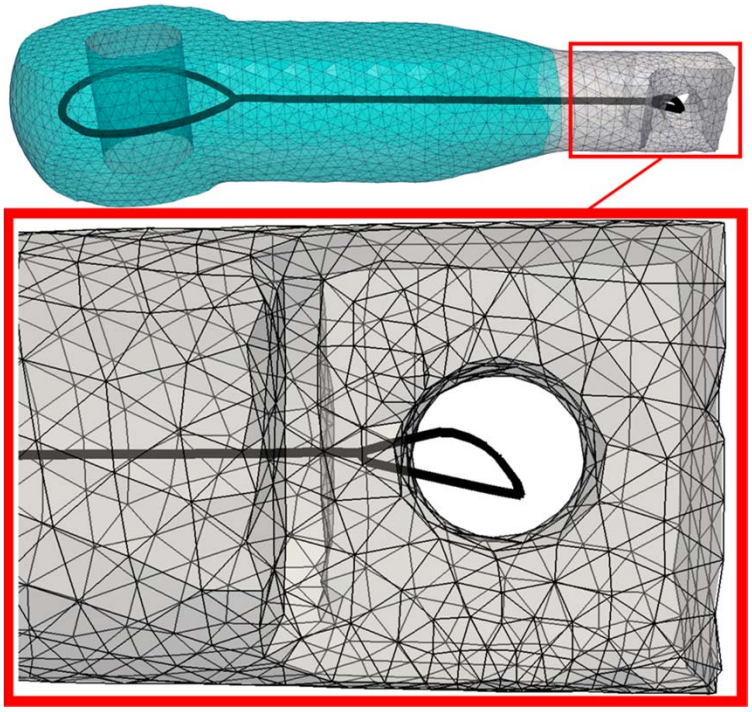
One Realization of the Pipeline

Segmentation through Skeletonization

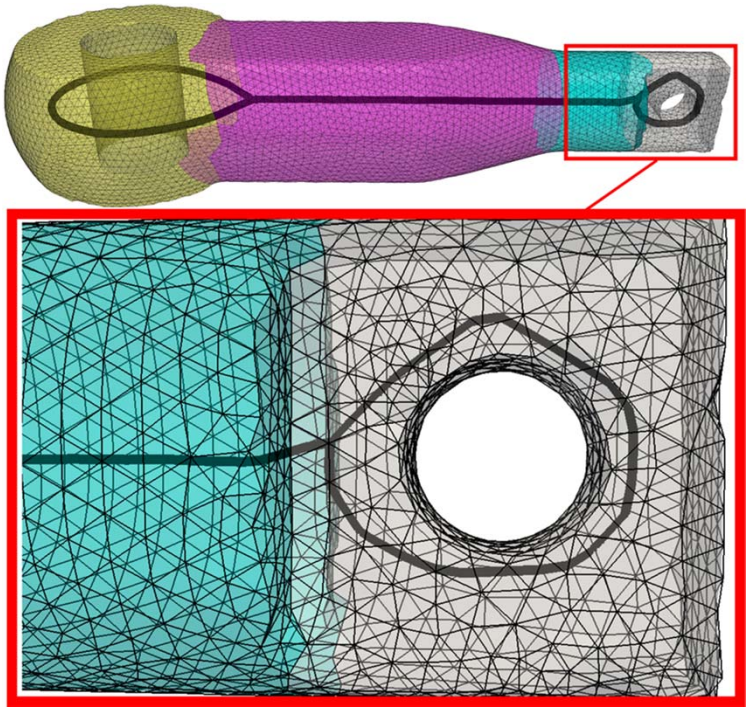


[Tagliasacchi et.al 2012 - Mean curvature skeletons]

Input Triangle Mesh Segmentation



5k triangles



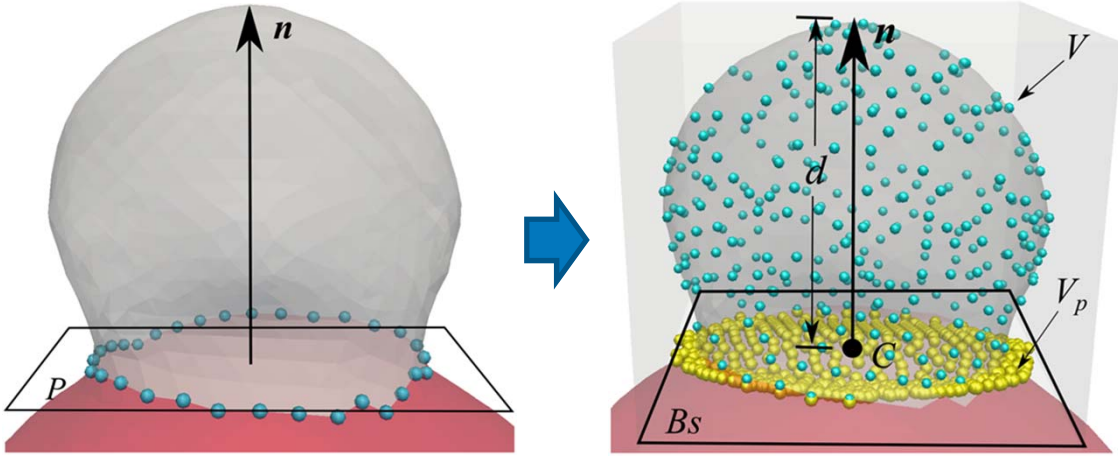
13k triangles

Identify Regions-of-Interest (ROIs)

Segments with feature size smaller than the average feature size of the entire mesh are candidates of ROIs.

The user can alsop manually select ROIs from these candidates, but this is not required.

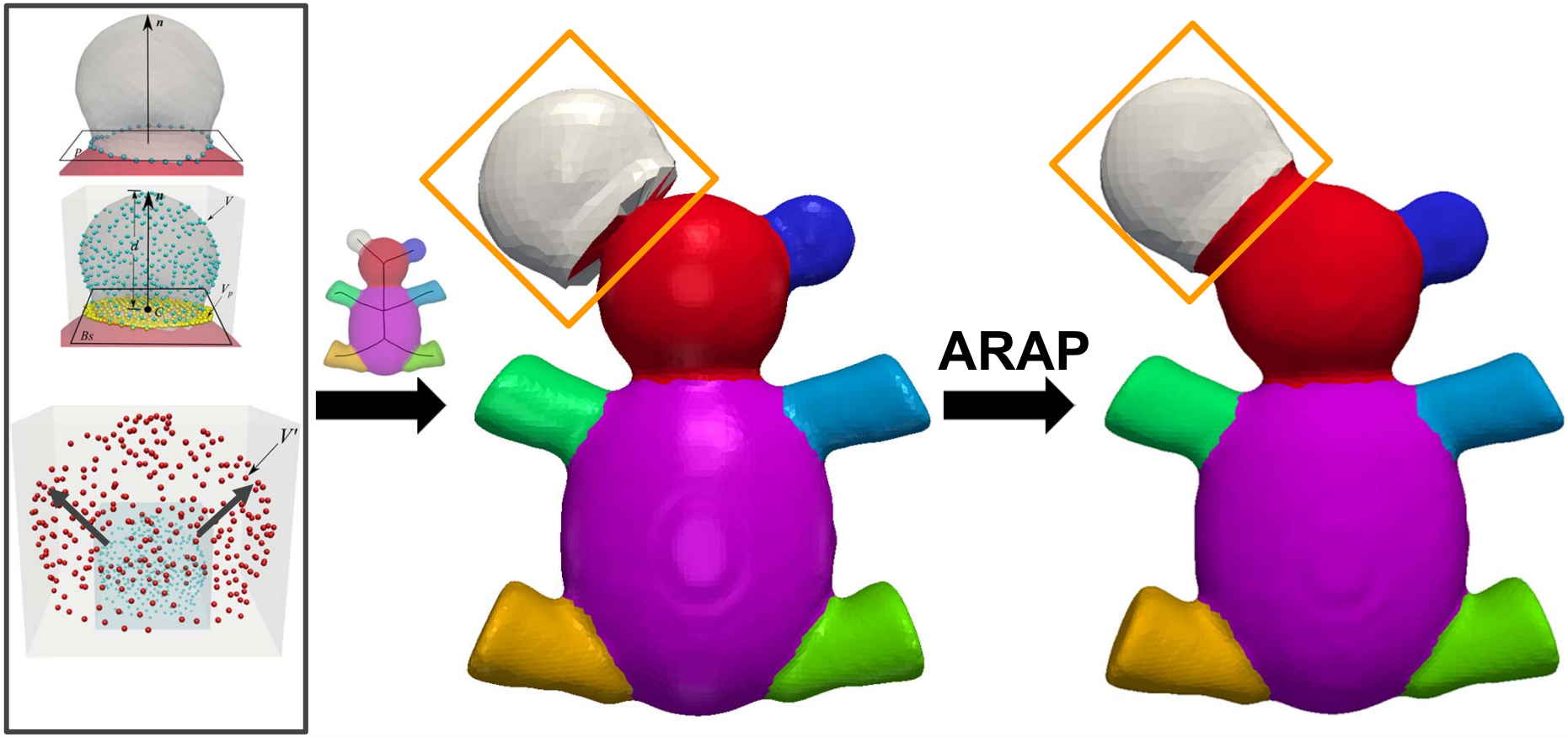
Magnification – Determine Bounding Box



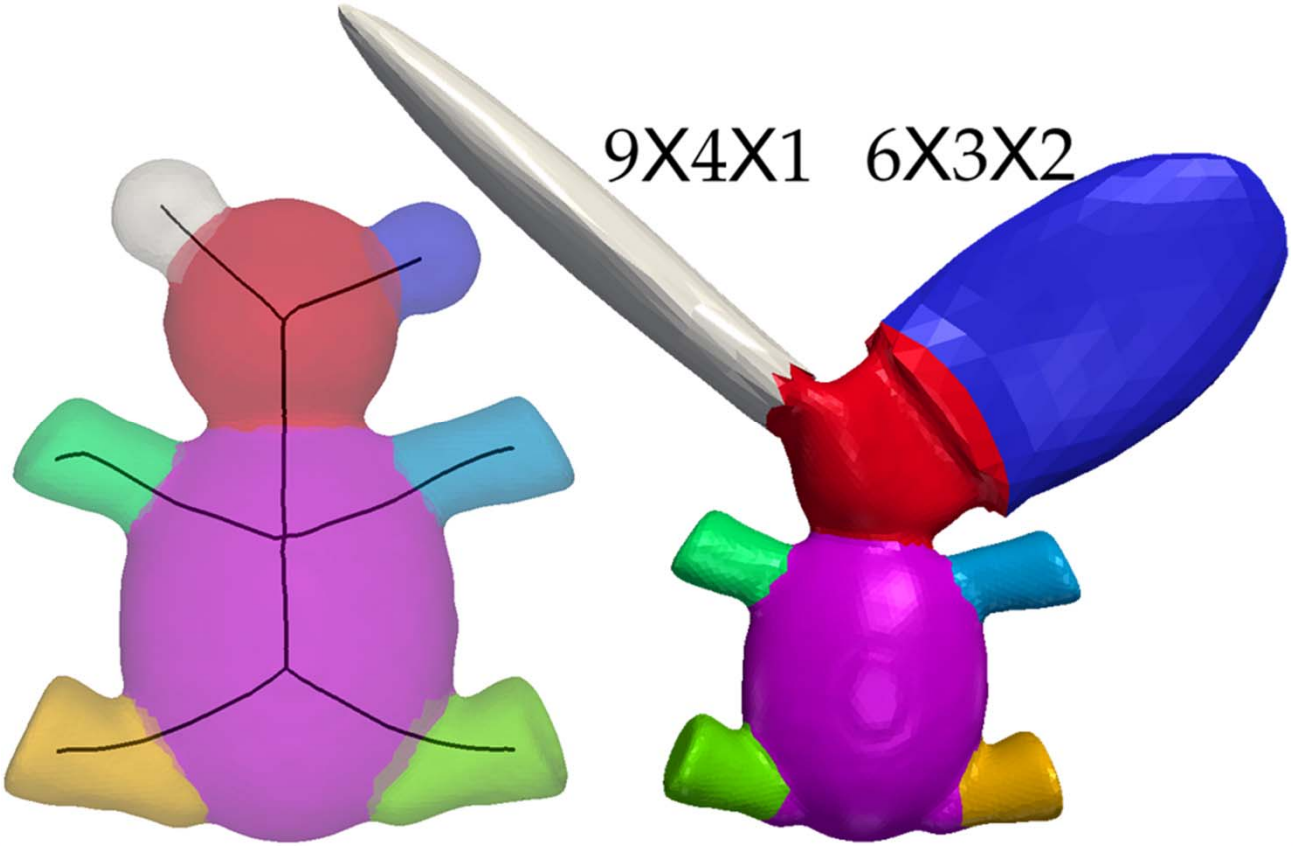
Scaling factor s_{avg}/s_{ROI}

Or user-specified

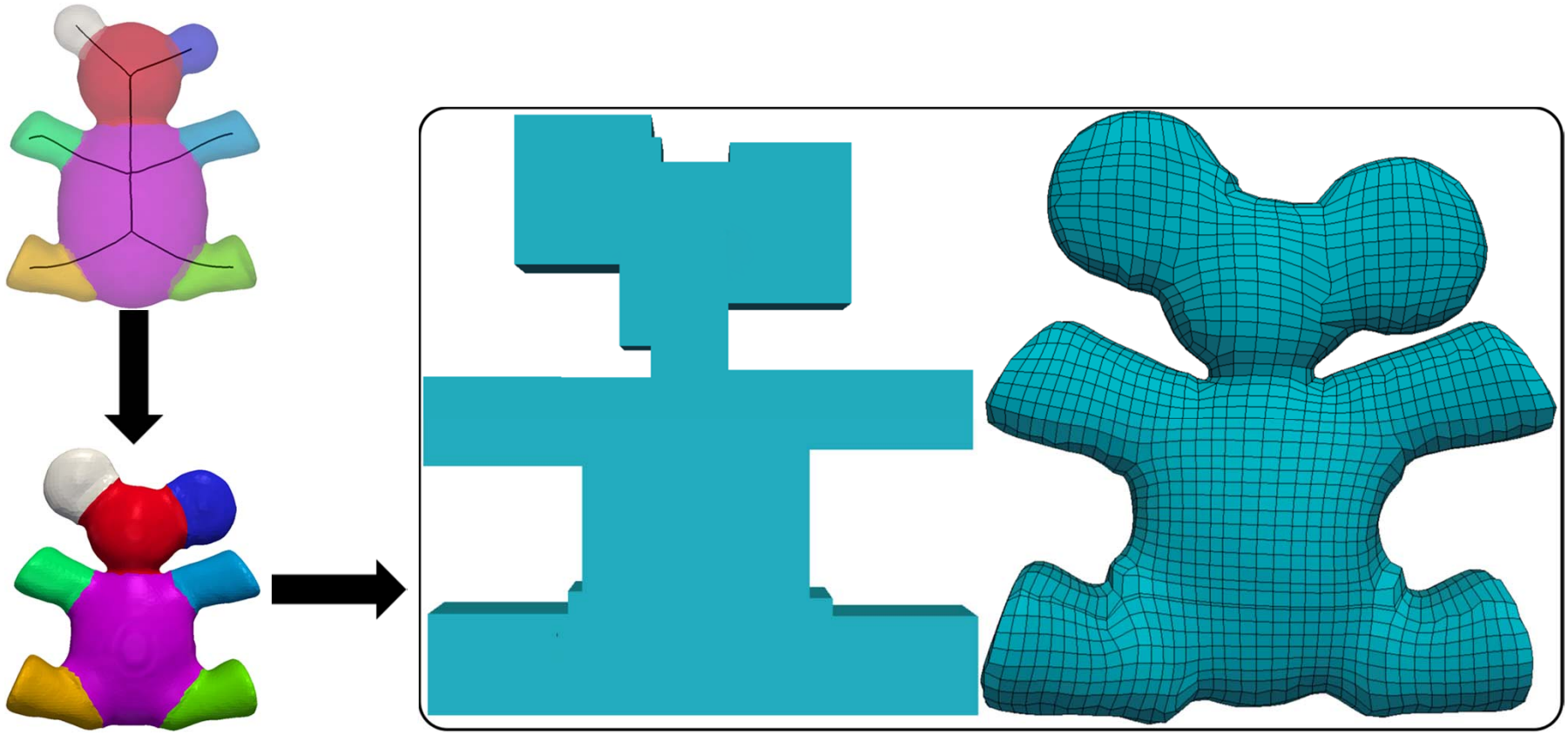
Magnification - Surface ARAP



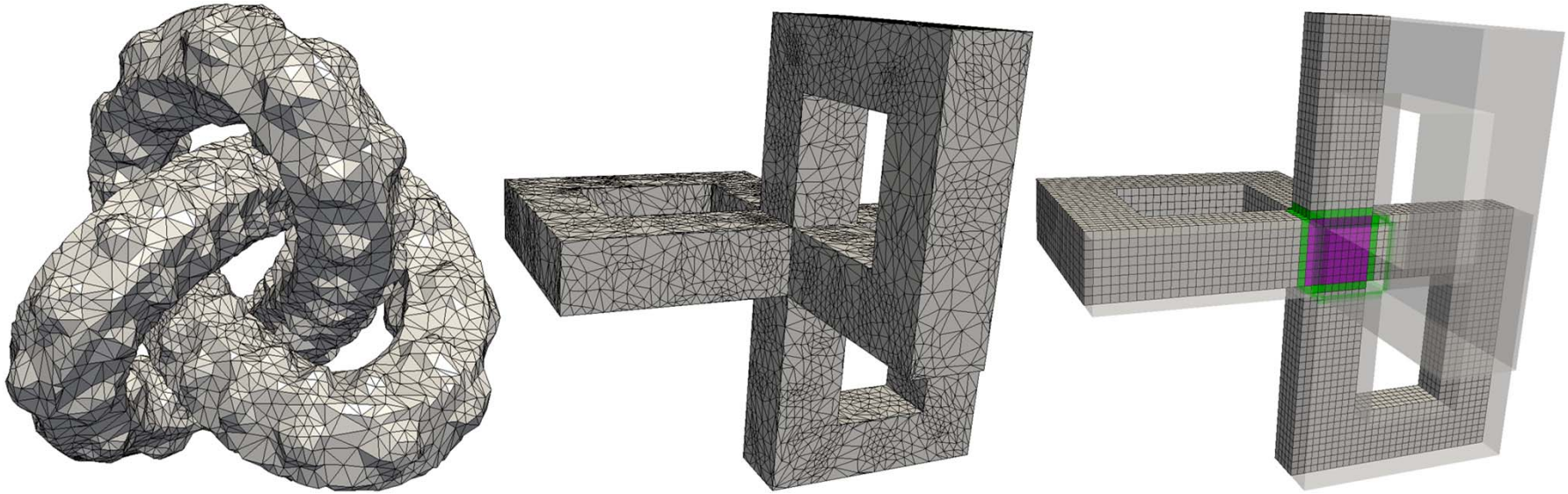
Anisotropic Magnification



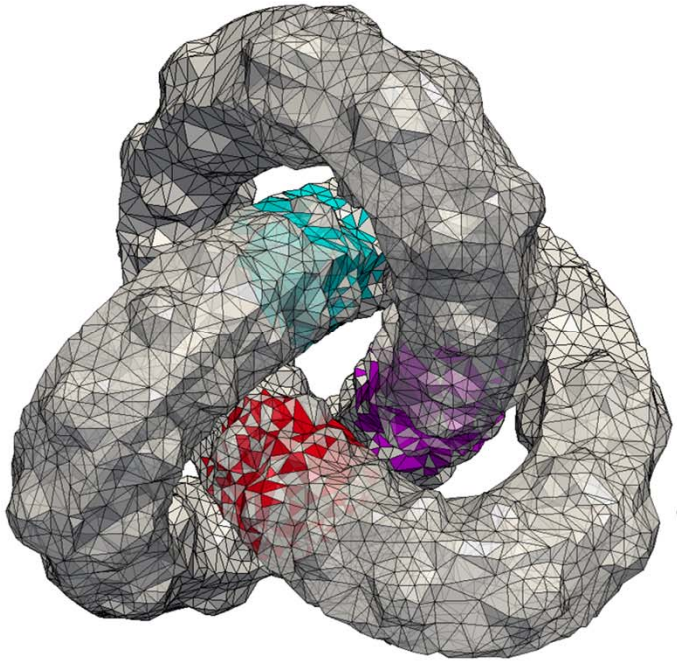
Hex-Meshing Using L1-PolyCube

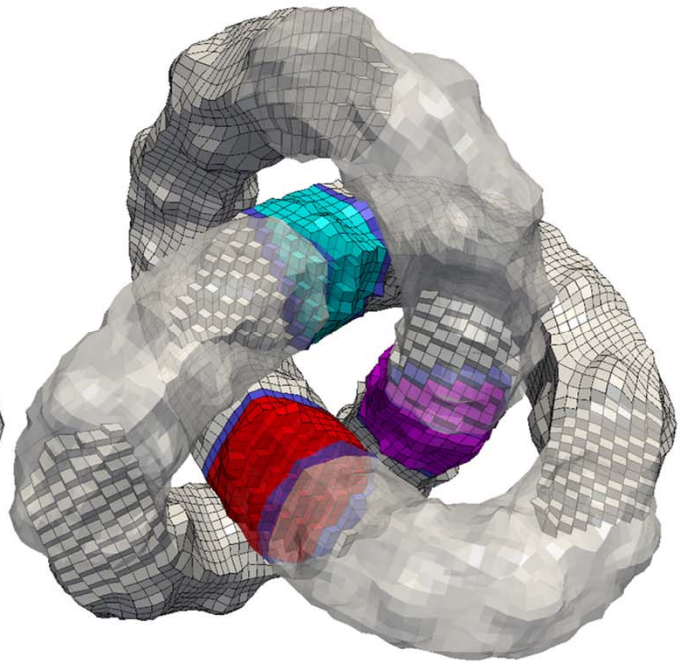


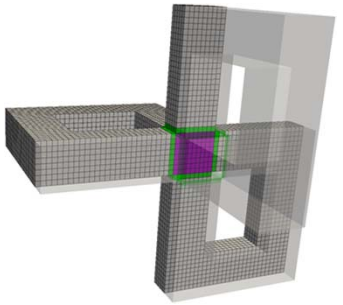
Issues of Using Polycube Methods



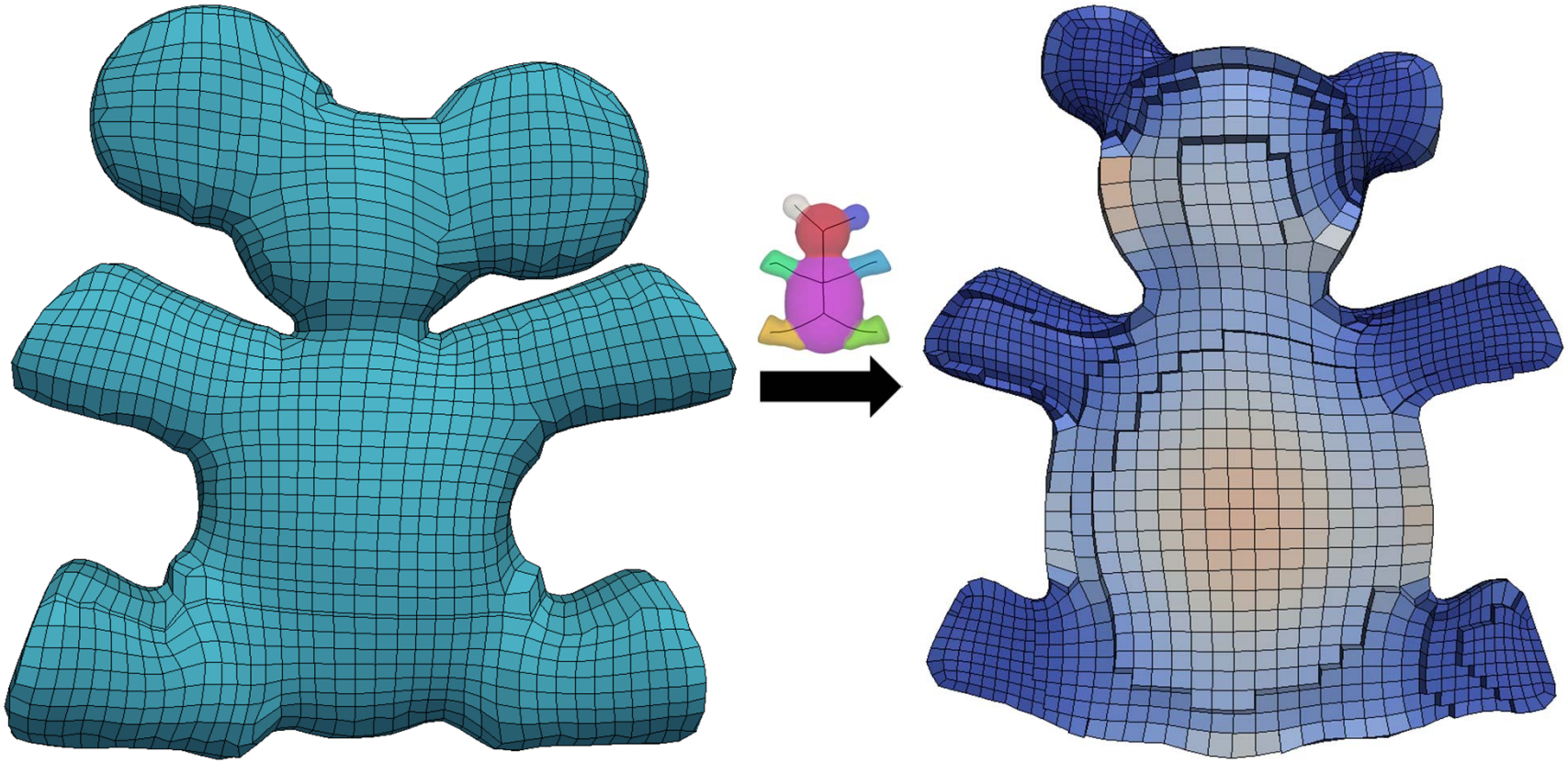
Issues of using polycube methods





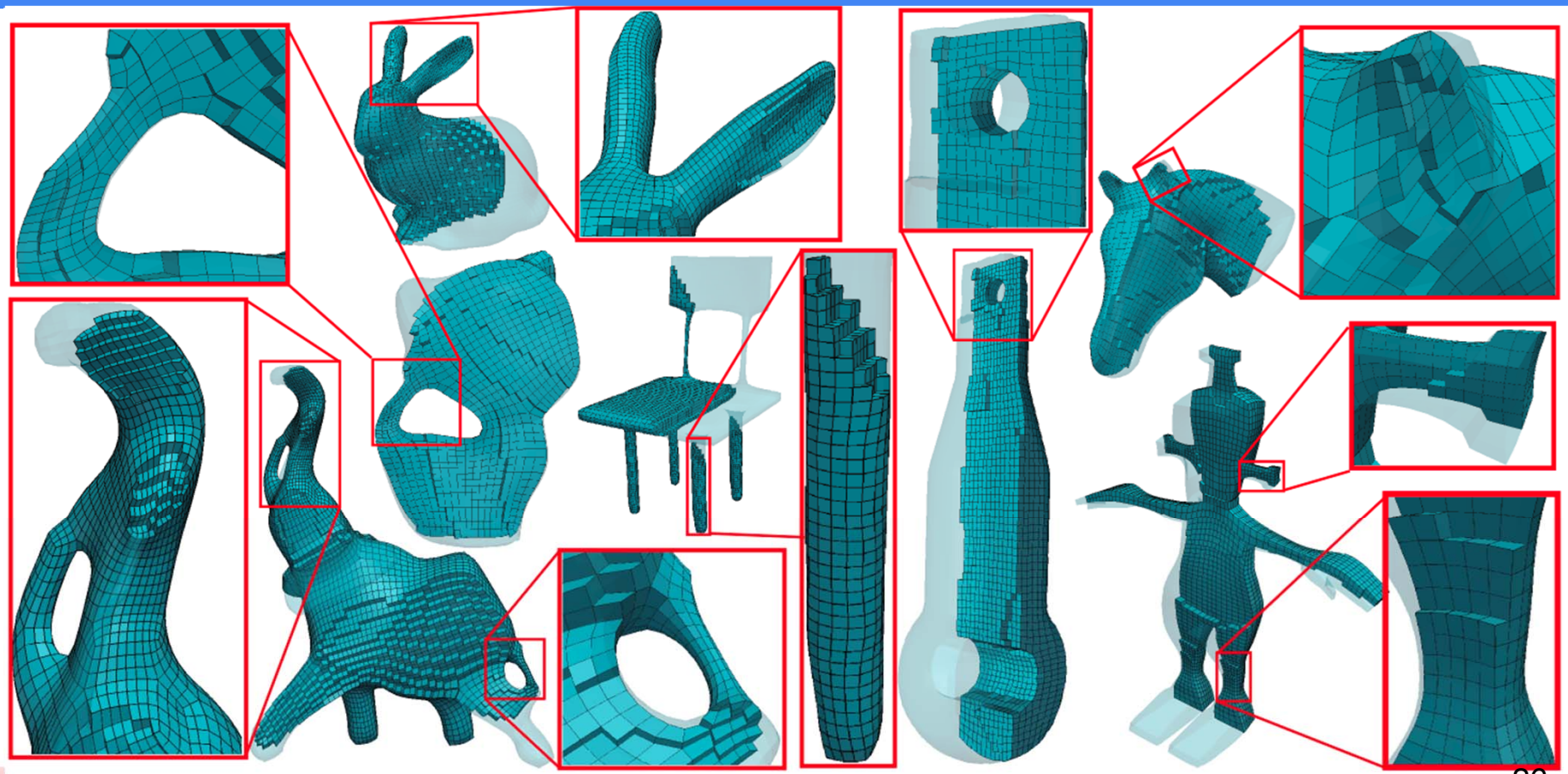


Mapping Back to the Original Space

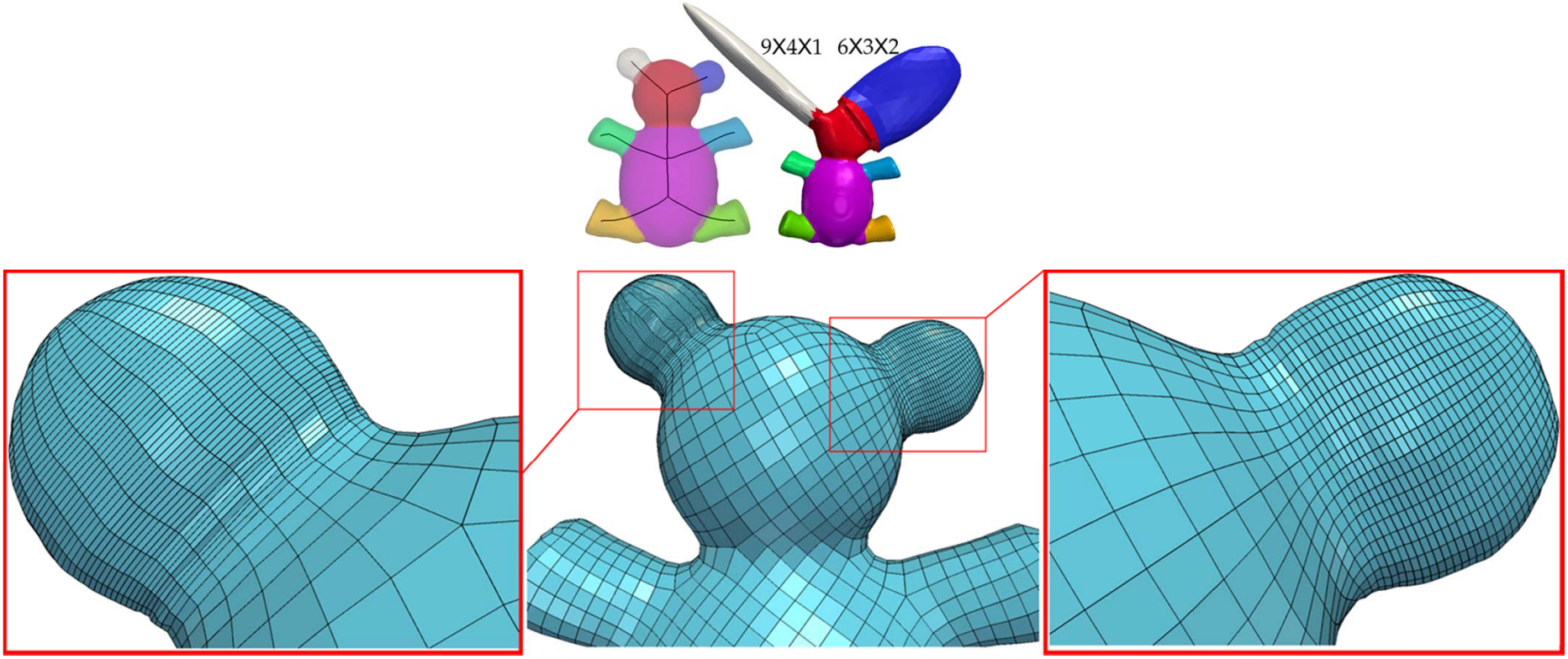


Results

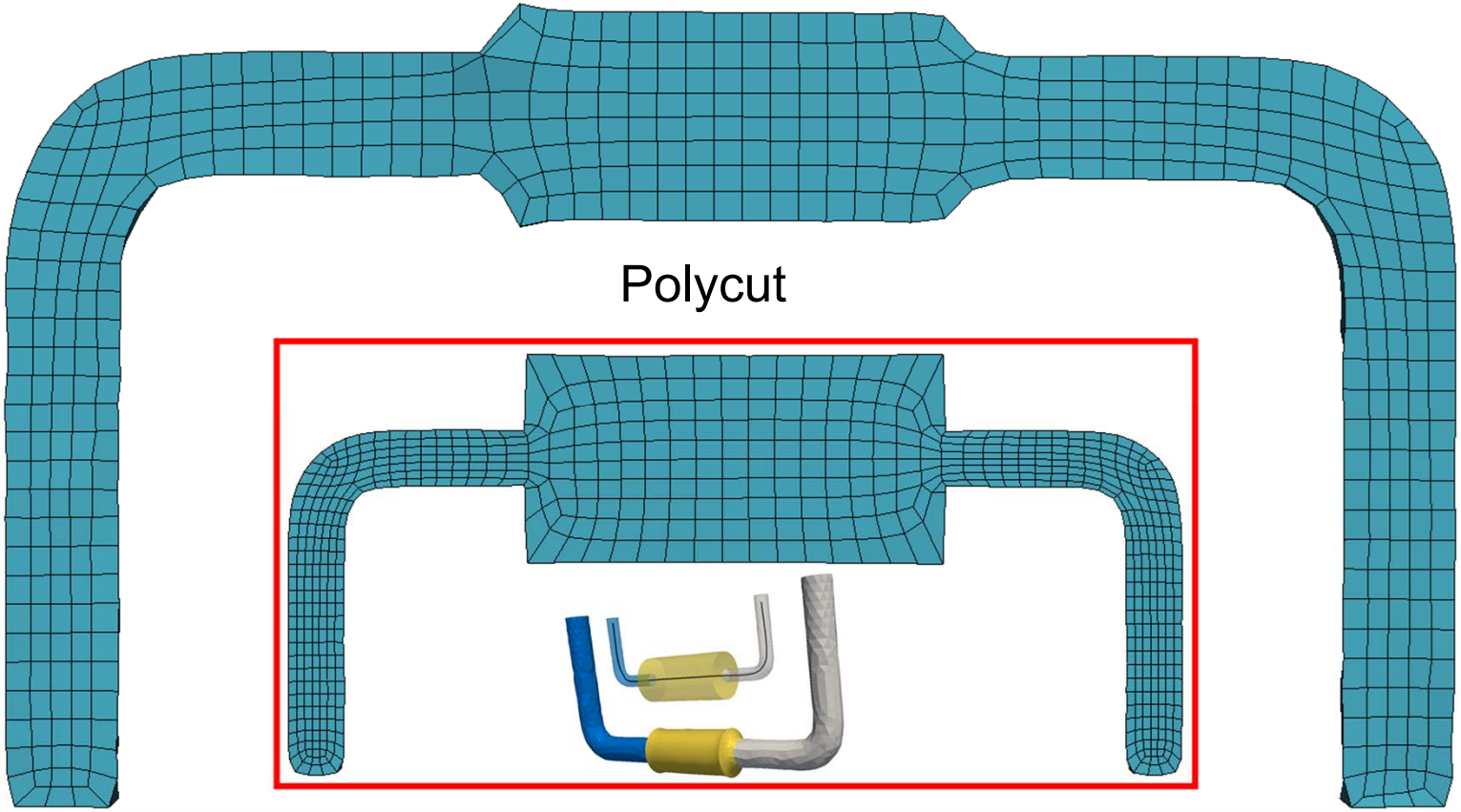
Results



Result - Anisotropic Meshes



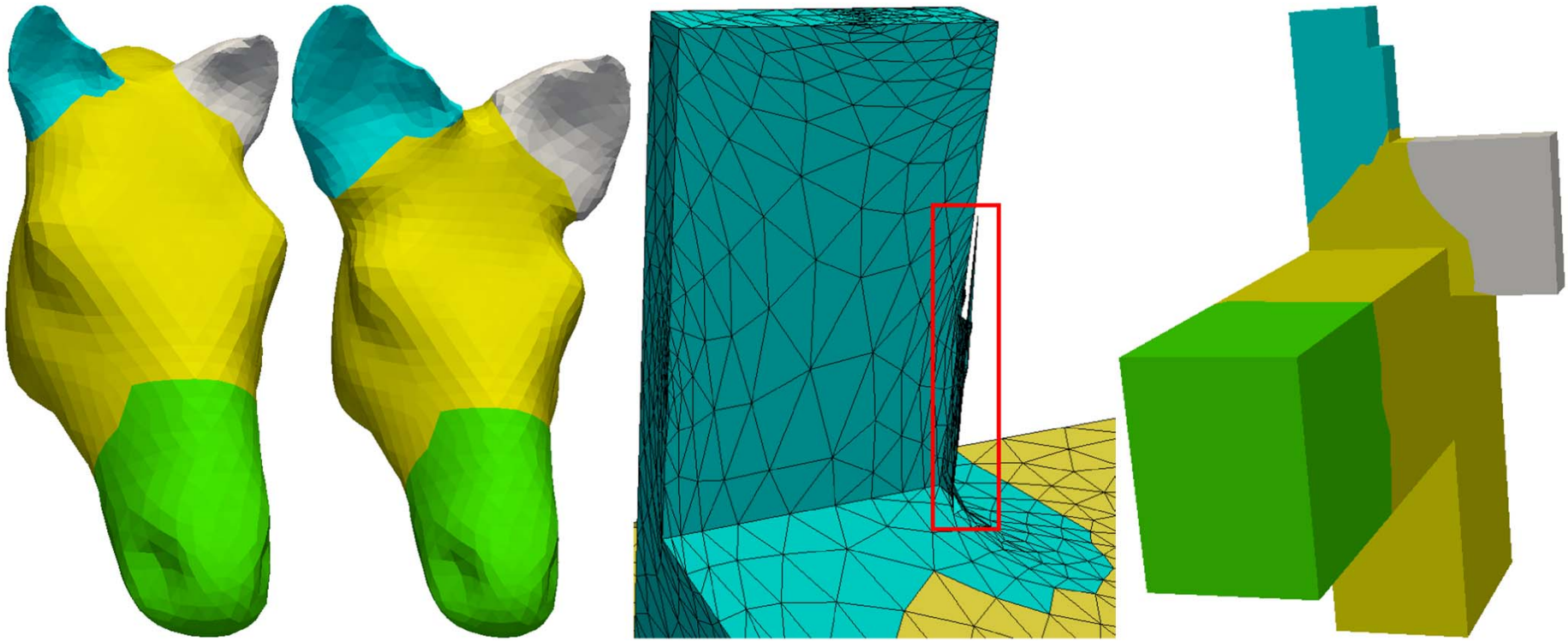
Result via Different Hex-Meshing Techniques



Result via Different Hex-Meshing Techniques

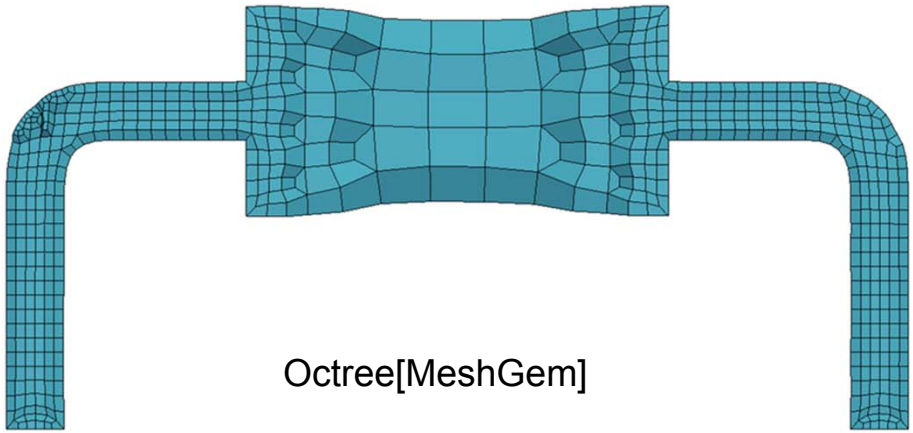
Closed-form
Polycube

Improve Polycubes

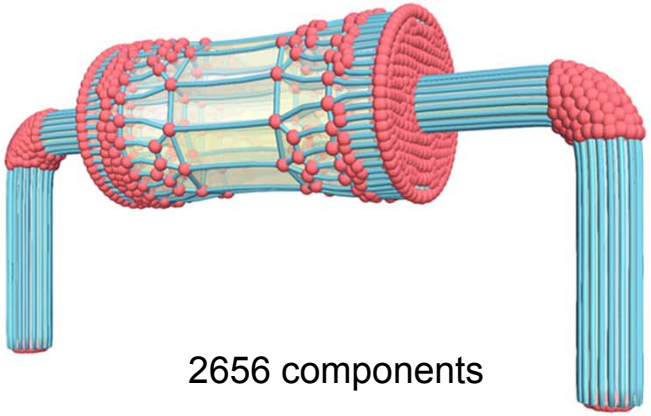


before after before after

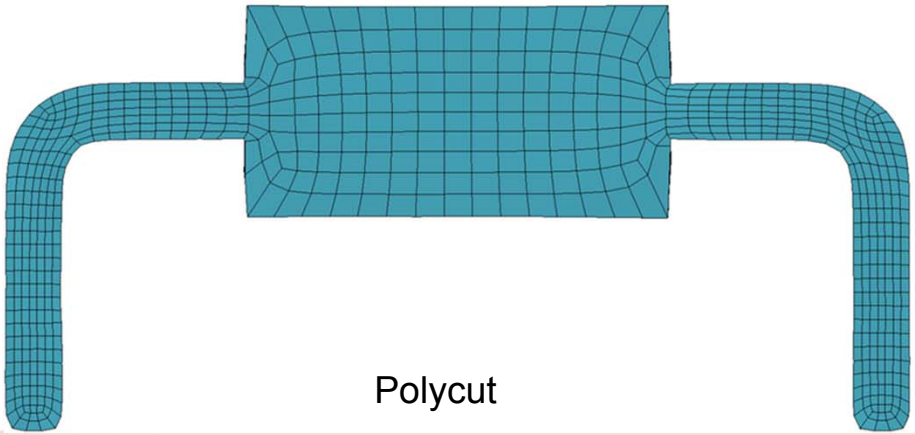
Singularities comparison



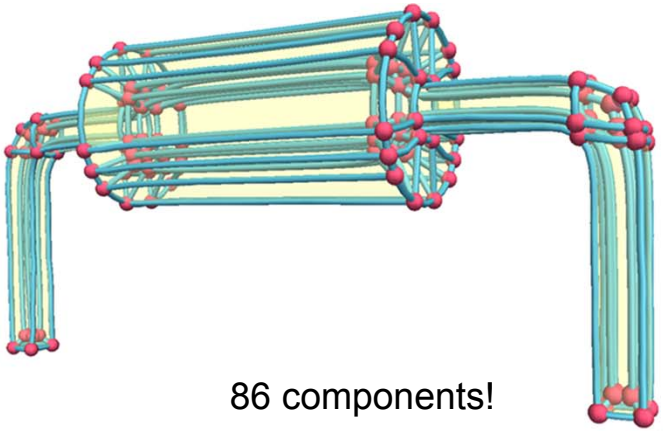
Octree[MeshGem]



2656 components

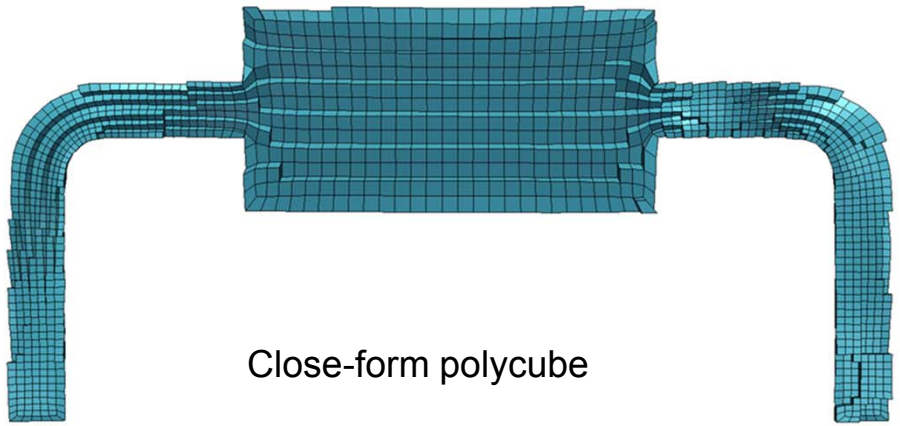


Polycut

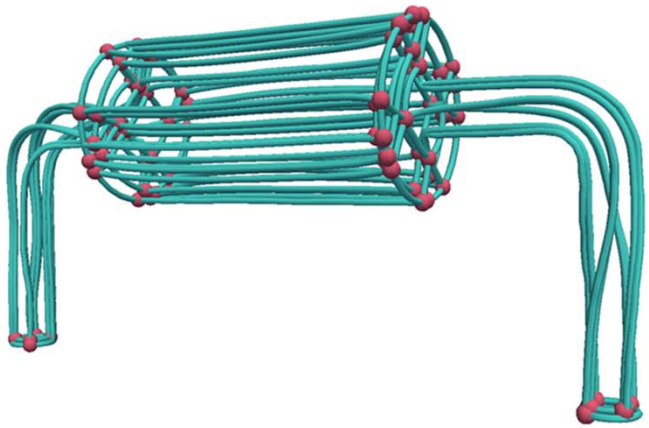


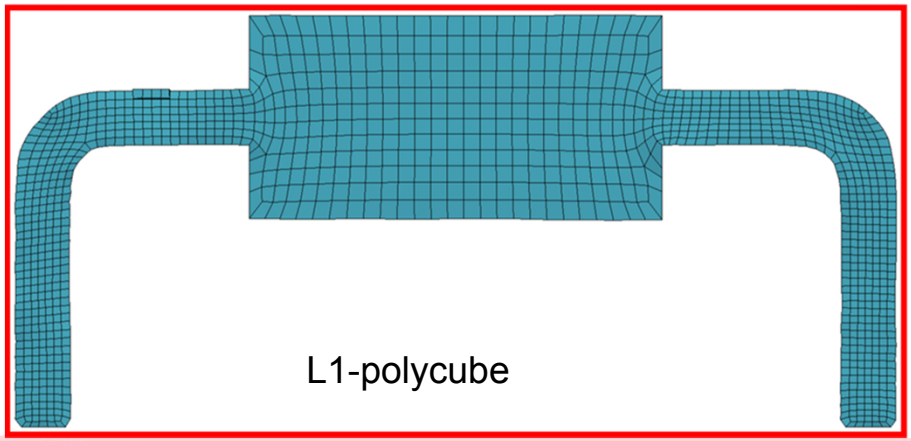
86 components!

Singularities comparison

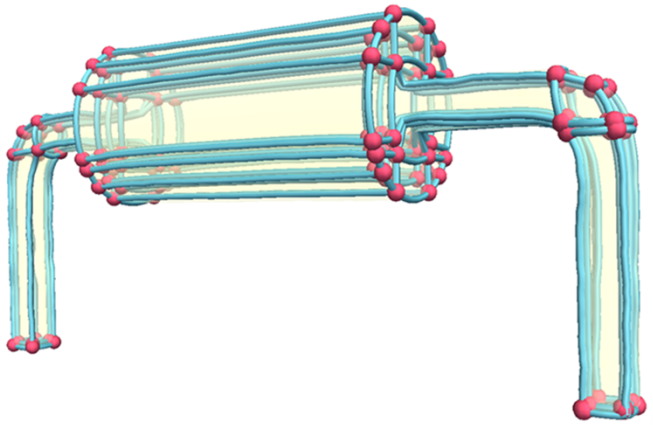


Close-form polycube





L1-polycube



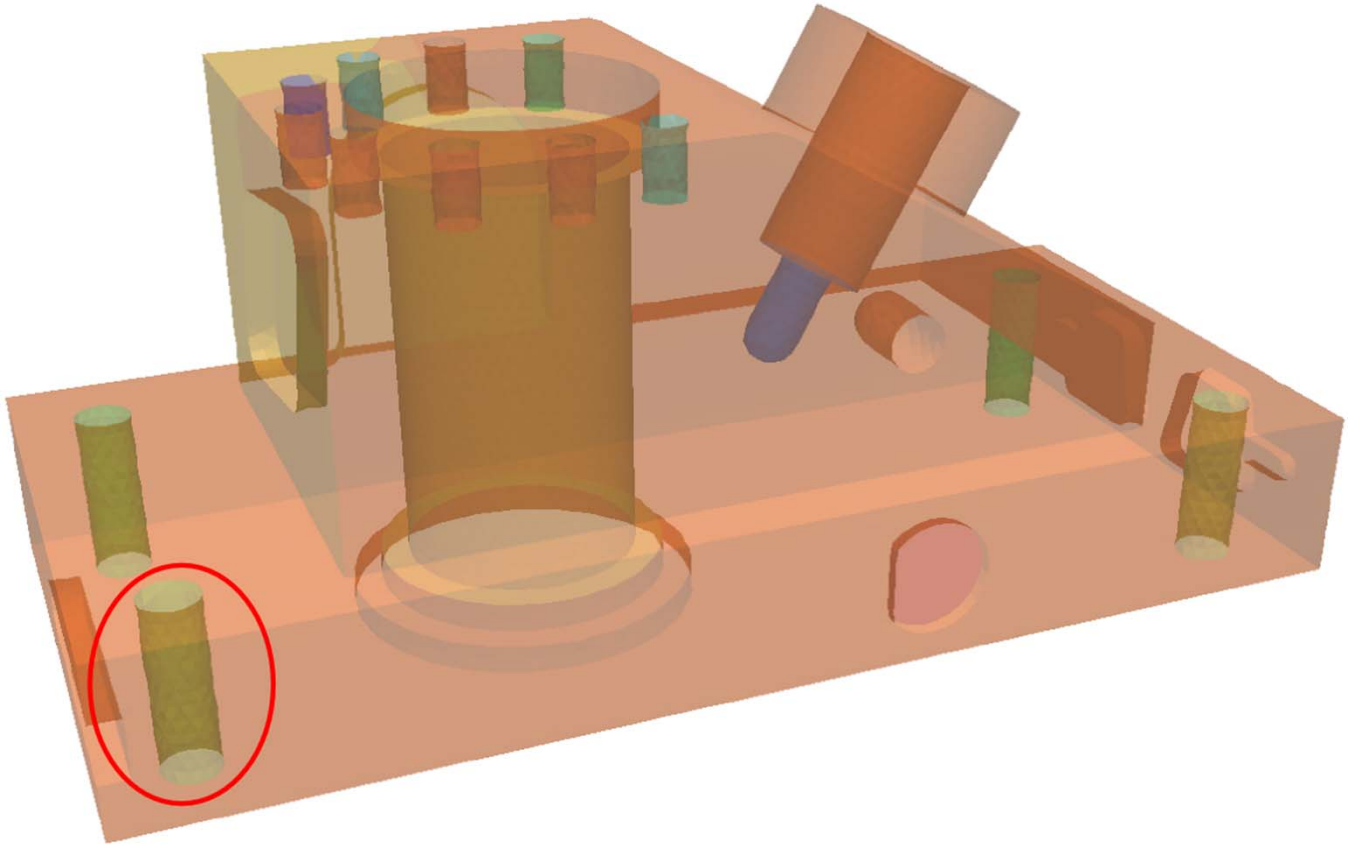
Performance

Model	#Tet	#Tri	#Seg	#Scal	#Sin	#Com	#Hex	MSJ / ASJ	H Dis	S Time	P Time	H Time
bumpycube	–	39936	–	–	60	121	16937	0.266/0.905	2.65	–	18m27s	16s
bear	98548	14626	8	2.2	188	467	7697	0.443/0.918	3.13	1.3s	209m31s	55s
bear*	10912	4588	–	–	140	104	10700	0.470/0.935	2.43	–	1m53s	10s
bunny	24169	7098	3	2.0	60	77	14571	0.275/0.900	2.92	0.5s	6m27s	19s
chair	14163	7490	8	2.0	160	266	9659	0.274/0.918	1.81	0.6s	2m43s	50s
chair^o	14163	7490	–	–	12005	35223	41664	0.076/0.864	1.08	–	–	1.2s
elephant	119321	24806	9	1.8	526	5827	23002	0.242/0.887	1.51	1.9s	193m10s	110s
horse	41831	7506	4	2.0	78	124	7523	0.259/0.876	6.14	0.5s	14m19s	21s
kitty	4521	1946	2	2.0	80	129	7124	0.291/0.887	2.56	0.2s	36s	6s
pipe	16285	4442	3	2.0	80	86	9045	0.419/0.933	1.59	0.2s	2m37s	12s
pipe^o	16285	4442	–	–	1847	2656	5571	0.092/0.790	3.92	–	–	1.1s
pipe*	16285	4442	–	–	80	69	7168	0.258/0.925	1.45	–	2m32s	14s
robot	27606	9032	10	1.9	196	598	8013	0.254/0.941	1.80	0.6s	6m30s	25s
rod	70178	13818	4	1.5	48	66	11448	0.063/0.932	1.14	1.2s	53m53s	48s

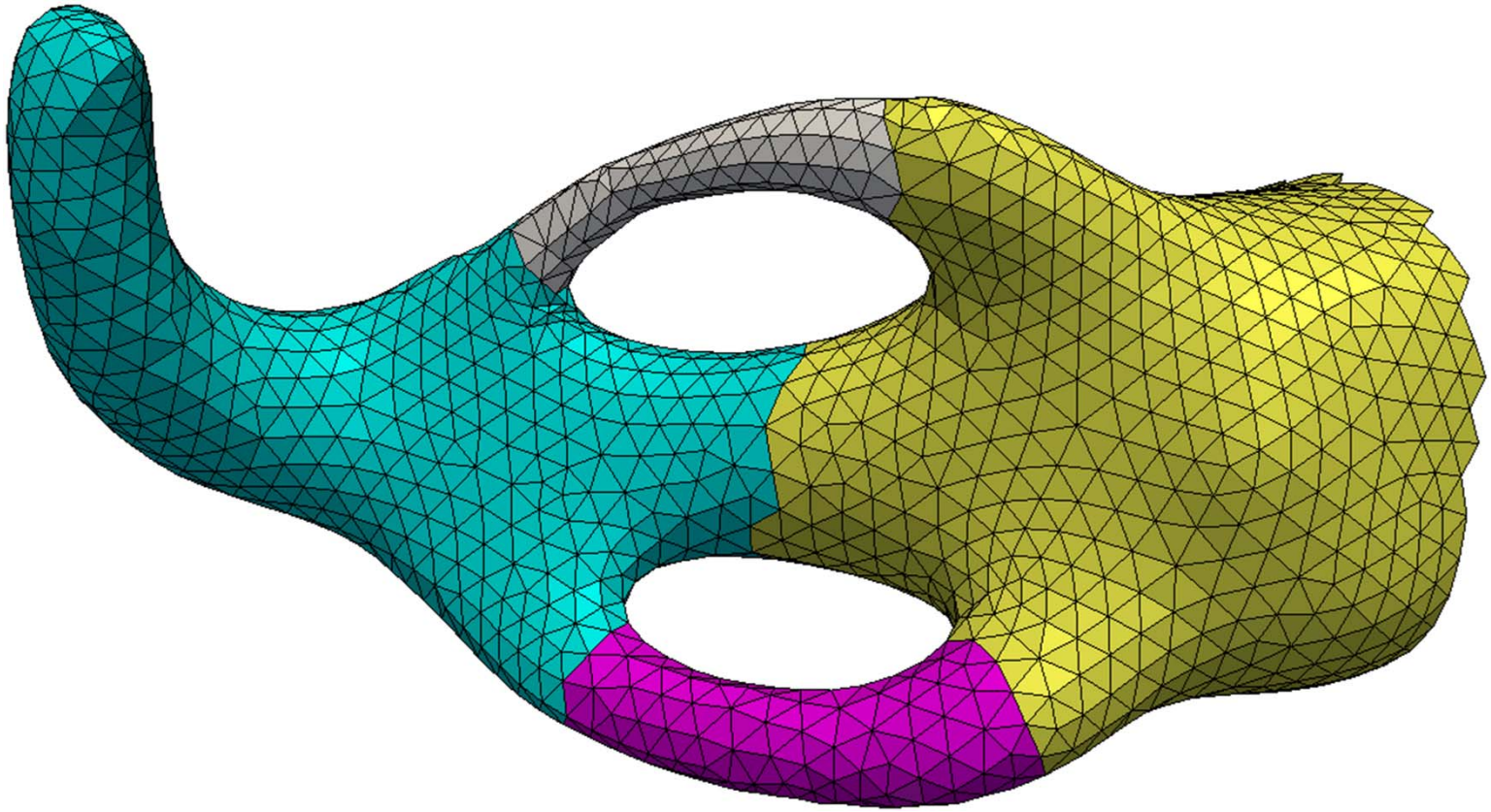
S Time, P time and H time show the timing for segmentation, L1-polycube construction, and hex-mesh extraction.

Limitations

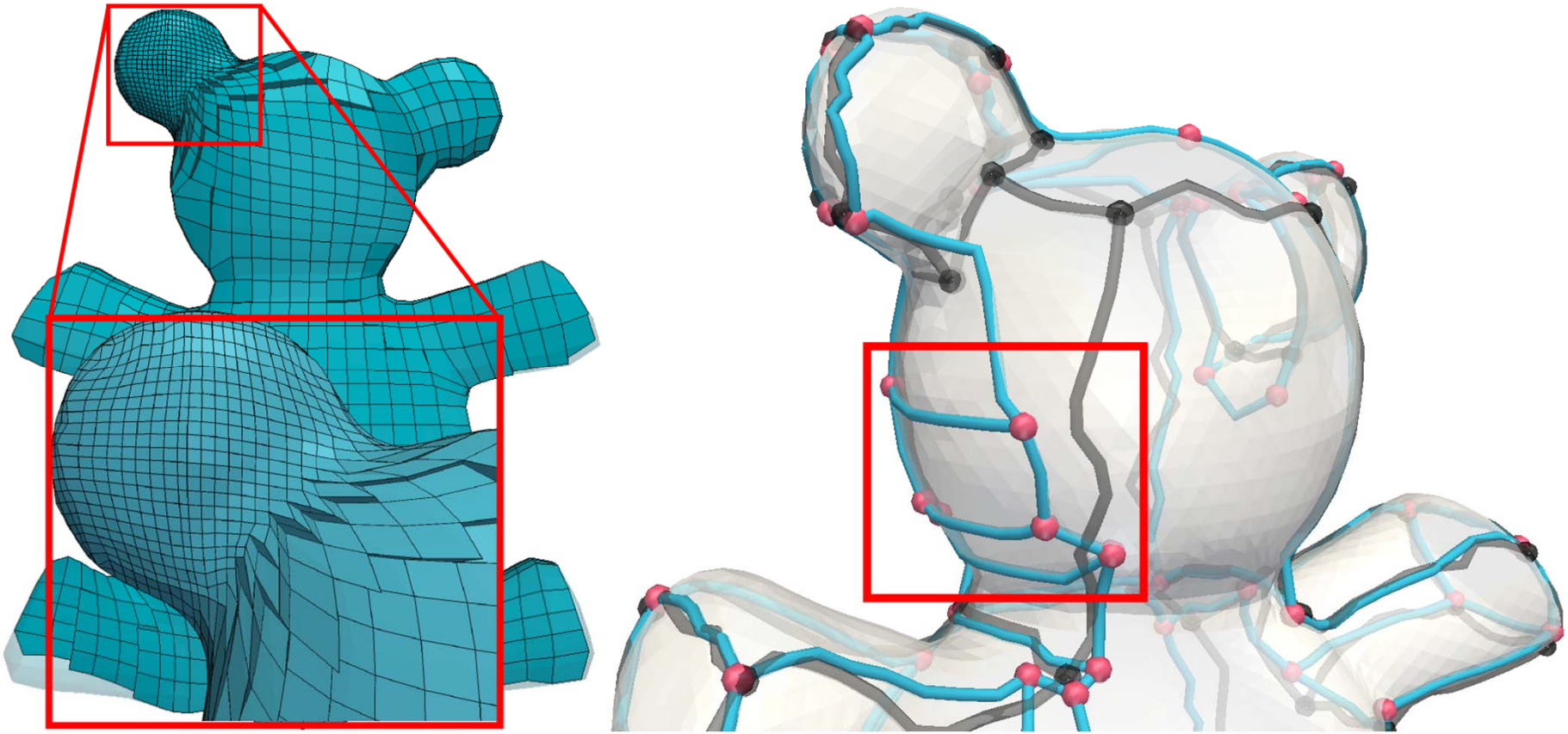
Limitation 1



Limitation 2



Limitation 3



Acknowledgment

We would like to thank Xianzhong Fang, Jin Huang, Marco Livesu, and Yang Liu for helping generate hex-meshes with their respective methods for comparison. This work was partially supported by NSF IIS-1553329.

Thank you for your attention!