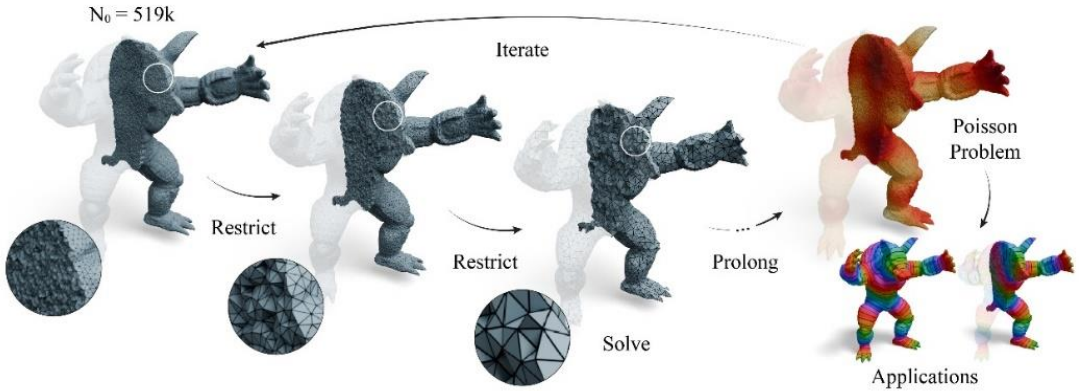
GravoTet

A Fast Multigrid Hierarchy Construction for Tetrahedral Meshes



Marcel Padilla[1], Ruben Wiersma[1], Tim Huisman[2], Jackson Campolattaro[2], Olga Sorkinge-Hornung[1], Klaus Hildebrandt[2]

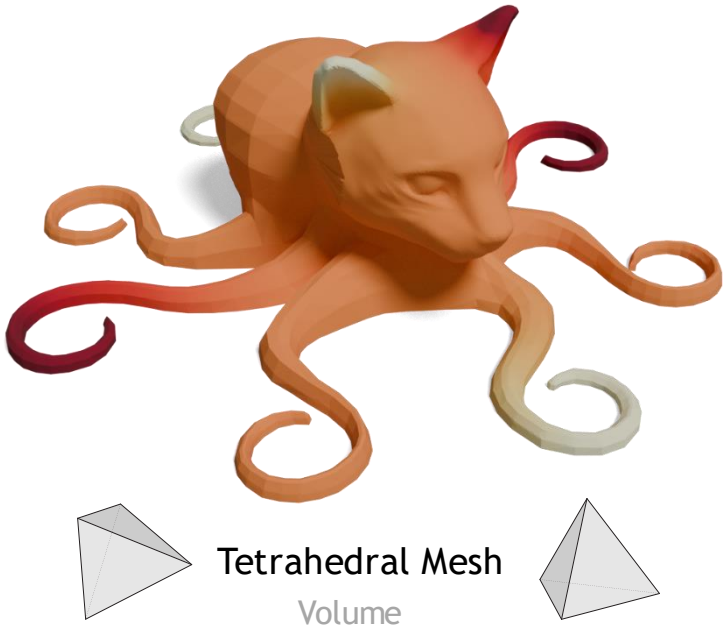
INTERACTIVE GEOMETRY LAB

[1]ETH Zürich, [2]TU Delft

SMI, July 2026

Motivation

Input:



$$\Delta u = 0$$

Heat equation

- Solving PDEs with Laplacians
  - Direct solving scales poorly
  - SOTA: CHOLMOD

Is there a faster way?

https://github.com/alecjacobson/sparse-solver-benchmark

Multigrid methods

$$Ax = b$$

Sparse symmetric matrix system

- Focus computation
- Hi-res = details
  - Use fast local methods
- Lo-res = global structure
  - Use global methods efficiently
- Communicate between levels
  - Residuals + smoothing

V-Cycle algorithm

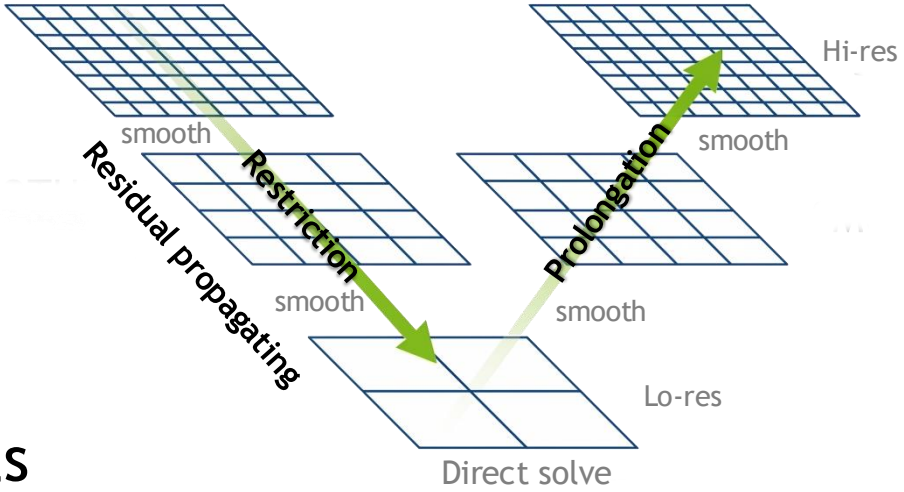


Image: Numerical Geodynamic Modelling - Taras Gerya

Multigrid methods

Matrix operatios:

$$P_l \in \mathbb{R}^{N_{l-1} \times N_l}$$

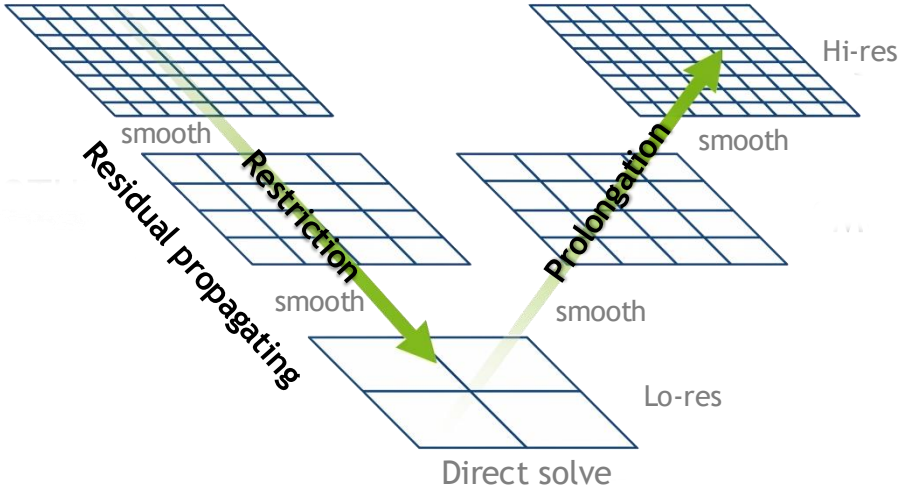
Prolongation matrix. Maps lo-res to hi-res

$$A^{l+1} = P_l^T A^l P_l$$

Matrix Restriction

Key ingredient: $P_1, P_2, \ldots$

V-Cycle algorithm



Multigrid methods

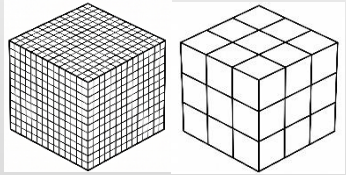
3D Volume Geometric Multigrid Methods

Input:

Tetrahedral Mesh

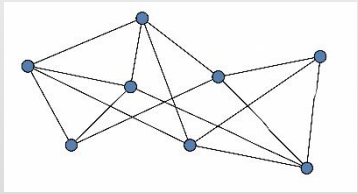
Boundary is a surface

Grid methods



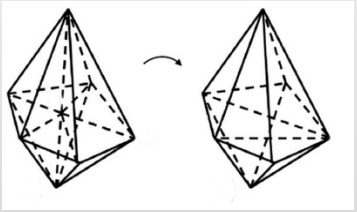
Poor boundary realization

Graph based methods



Slow convergence

Tet methods: edge collapse



Very slow construction

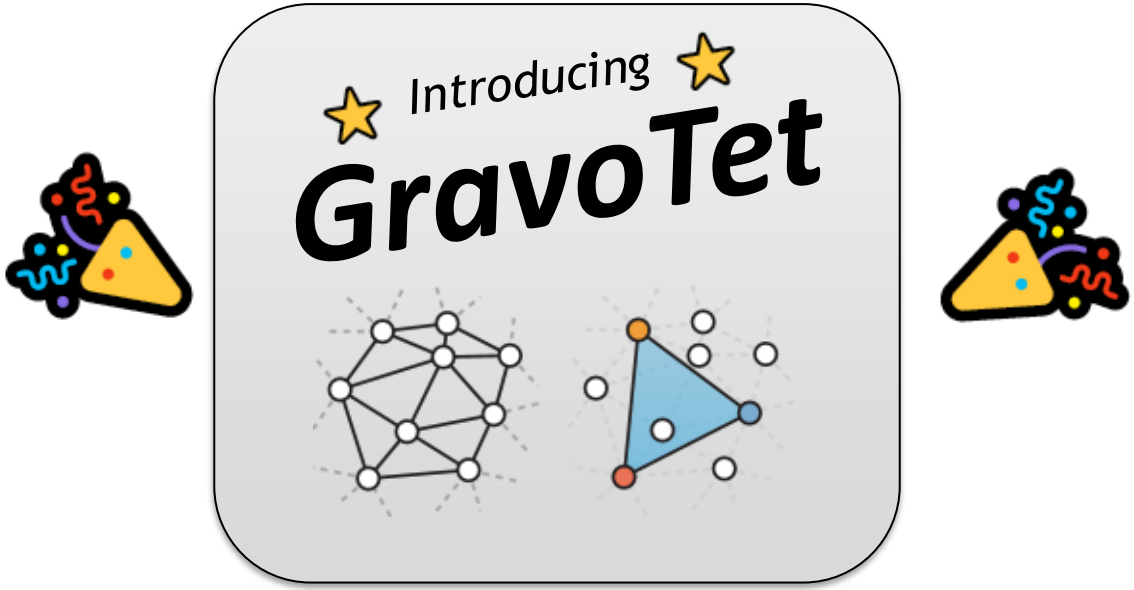
Can we get fast construction of P_l with good convergence? 🔮

Multigrid methods

Input:

Tetrahedral Mesh

Boundary is a surface



GravoTet

Introducing **GravoTet**

Graph **Vo**ronoi **Tet**rahedral Multigrid Method

fast & effective coarsening strategy

Prioritized fast disk sampling

Select vertices of coarser level

Surface Voronoi cluster + connect

Preserve the boundary. Approximately Delaunay.

Interior Voronoi cluster + connect

Connect the interior. Approximately Delaunay.

Prolongation construction

$[P_1], [P_2], [P_3], \ldots$

Build P_l using barycentric weight.

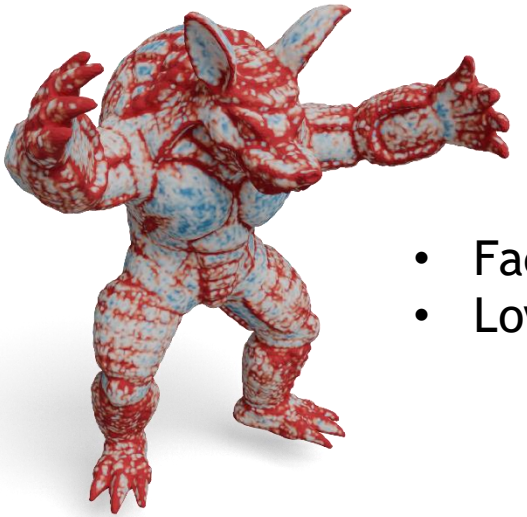
GravoTet

Prioritized fast disk sampling

Select vertices of coarser level

📐 Compute and sort vertices by features

Boundary features to preserve!



- Face-star normal variation
- Low in flat areas.

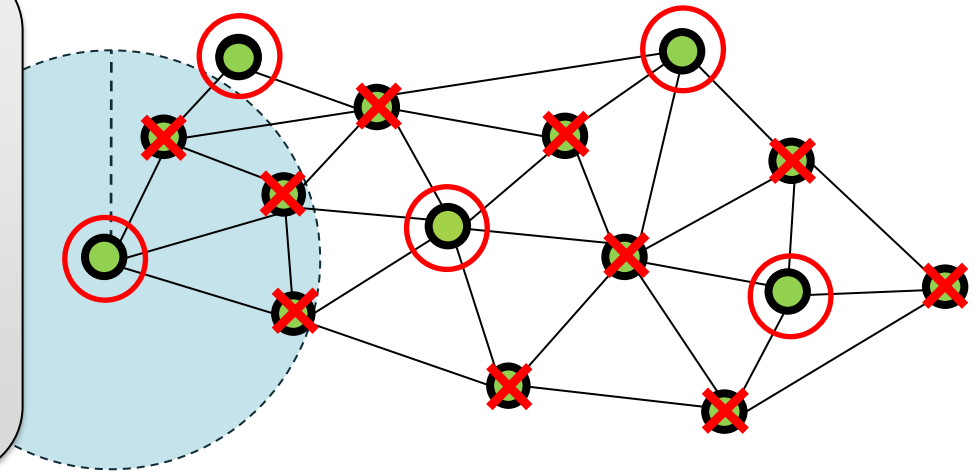
GravoTet

📐 Compute and sort vertices by features

Boundary features to preserve!

Prioritized fast disk sampling

Select vertices of coarser level



- Exclude points that are too close
- Exclusion check 2-ring neighbours.
  - Linear time

💡 The resulting vertex list sorted by parent feature!

(no resorting needed)

GravoTet

Core idea

Surface Voronoi cluster + connect

Preserve the boundary.
Approximately Delaunay.

Voronoi diagram ⟷ (dual) Delaunay

Graph Voronoi diagram ⟷ (≈ dual) Approximately Delaunay

Faster

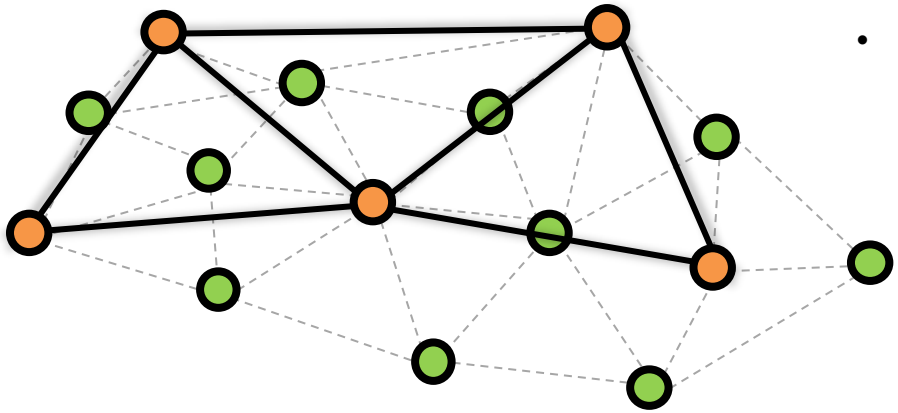
GravoTet

Surface Voronoi cluster + connect

Preserve the boundary.
Approximately Delaunay.

👉 Cluster by Voronoi conectivity

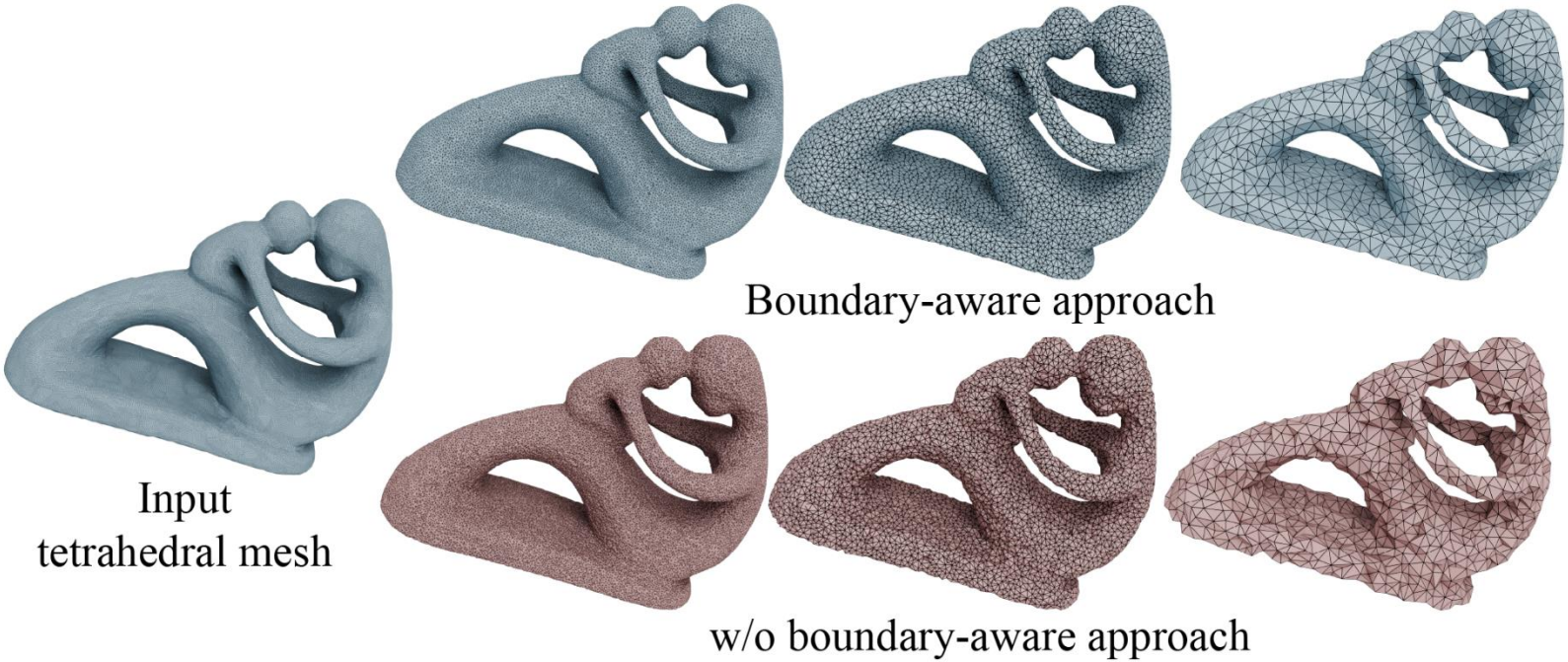
Assign fine vertices to the graph-nearest coarse vertex representative



- If two fine vertices with different coarse representative vertices are connected:
  - Connect the coarse vertices
  - Fast 🐇
  - *Not Delaunay*

🚨 Used ONLY the surface vertices! To preserve boundary

Surface first!



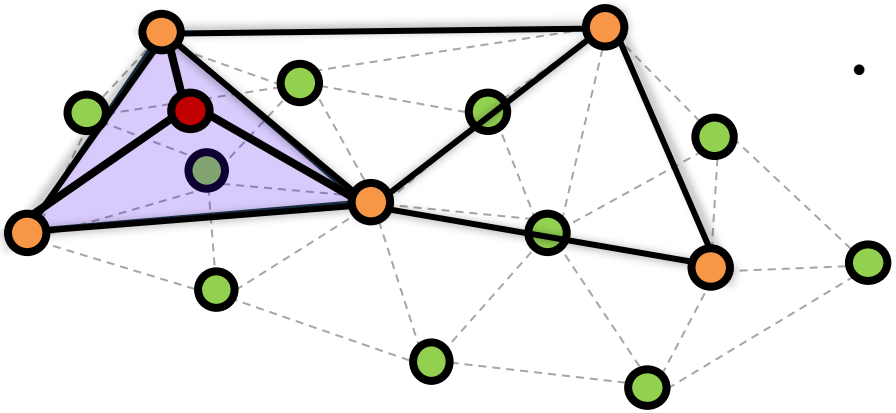
GravoTet

Repeat the same step to the interior!

The surface connectivity remains intact.

Interior Voronoi cluster + connect

Connect the interior. Approximately Delaunay.



- If two fine vertices with different coarse representative vertices are connected:
  - Connect the coarse vertices
  - *Not Delaunay*

Now **ALL** the coarse vertices are connected

GravoTet

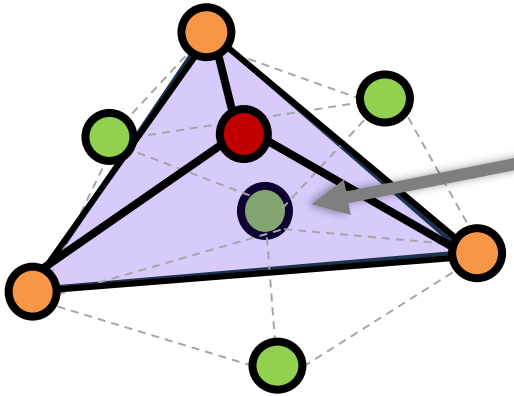
Compute tet weights

Or project to triangles, lines, points. The surface connectivity remains intact.

Prolongation construction

$[P_1], [P_2], [P_3], \ldots$

Build P_l using barycentric weight.

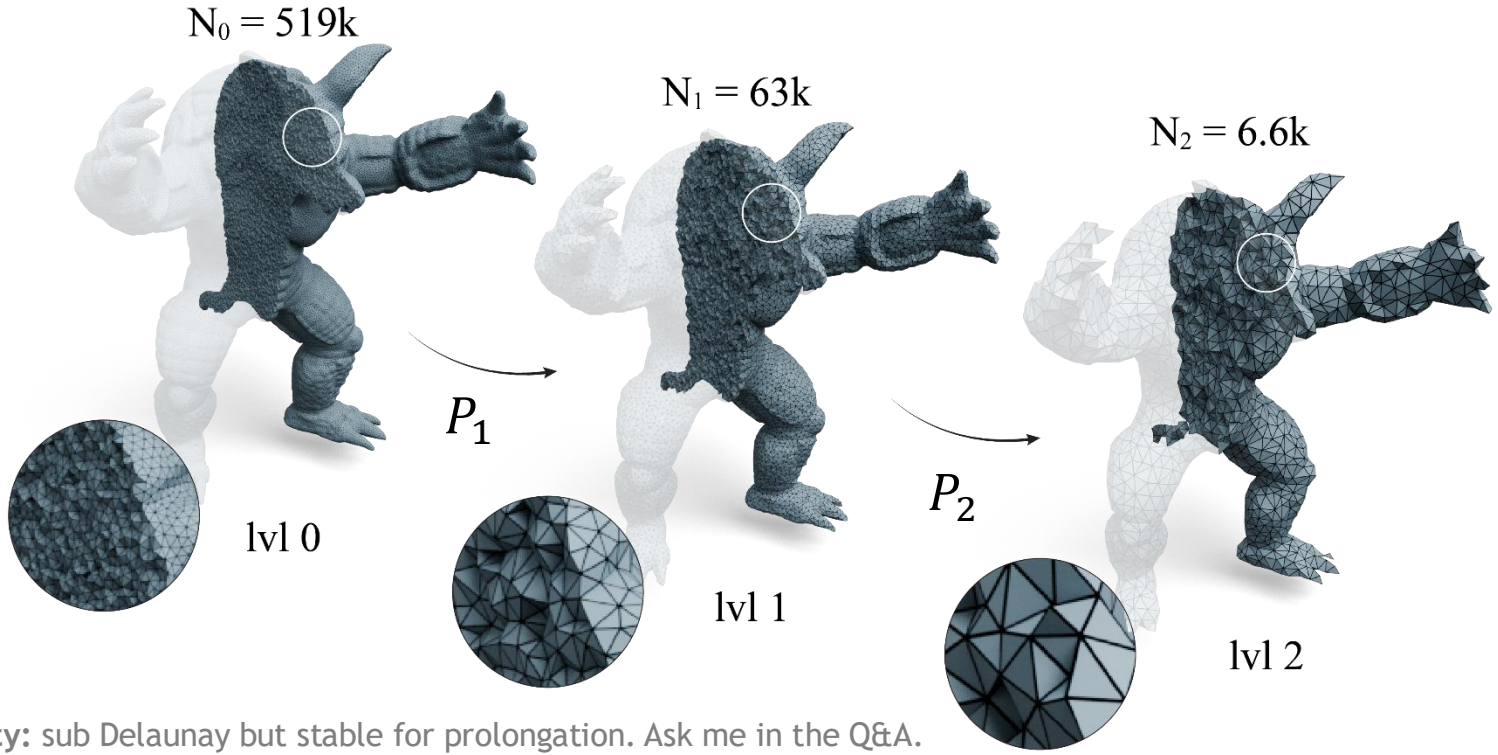


For the fine points:
- Identify containing tet
- Compute barycenter weights for P_l

P_l has at most 4 non-zero entries per row.

⚠ Tetrahedra may overlap. And that is ok! ⚠

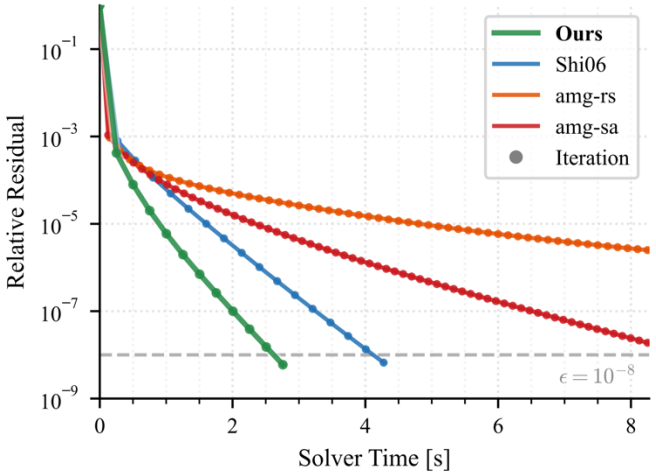
Resulting Mesh Hierarchy

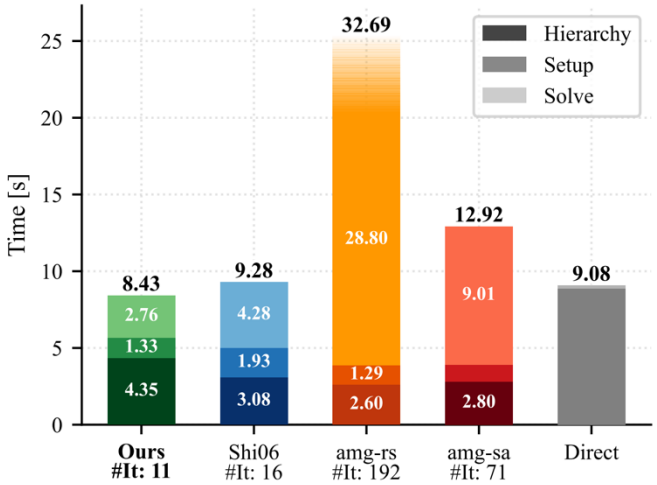


Mesh quality: sub Delaunay but stable for prolongation. Ask me in the Q&A.

Armadillo *Poisson* Problem

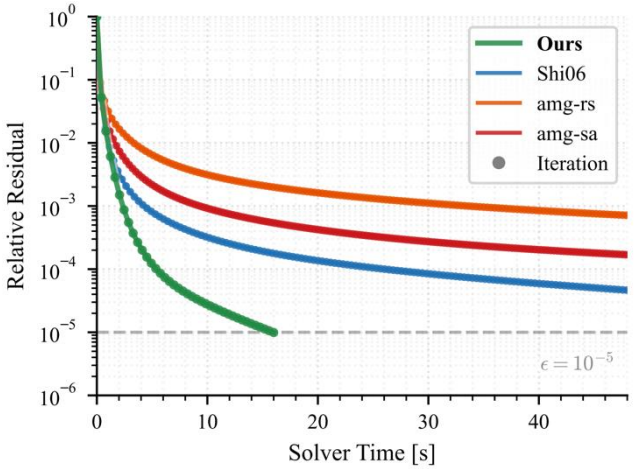
Input: 519k vertices

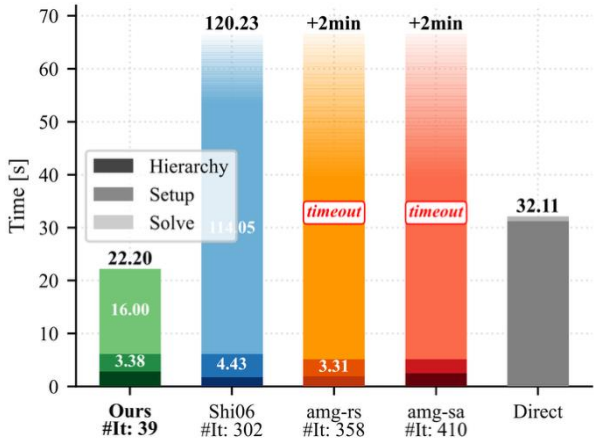




Armadillo ***Biharmonic*** Problem

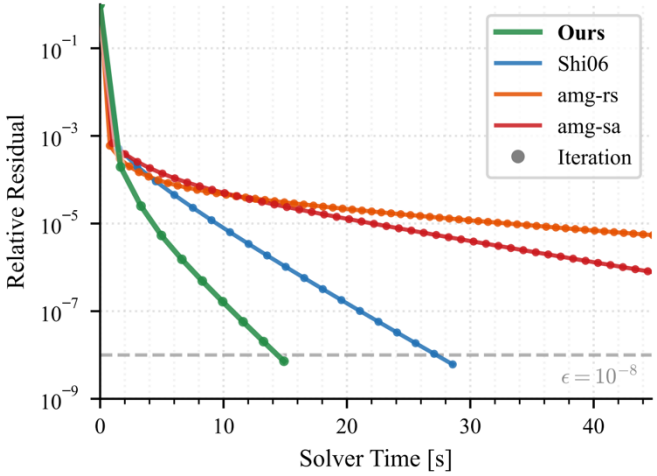
Input: 519k vertices

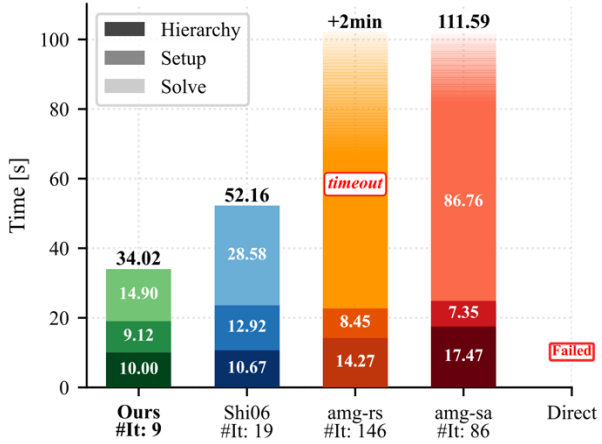




Fertility *Poisson* Problem

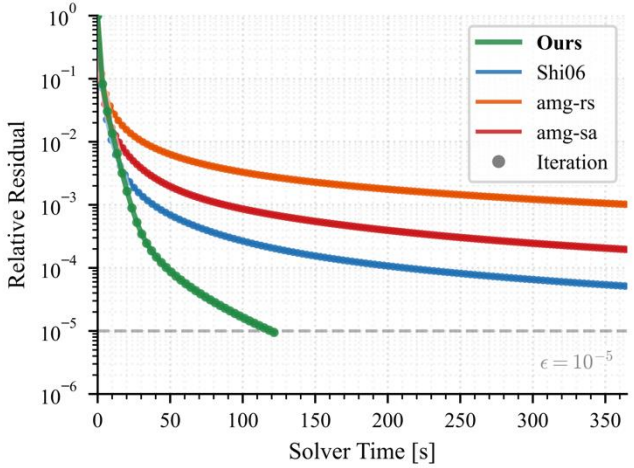
2.8M vertices

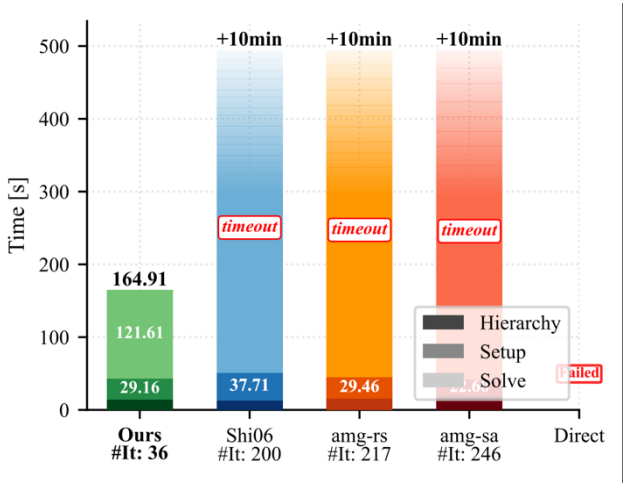




Fertility ***Biharmonic*** Problem

2.8M vertices



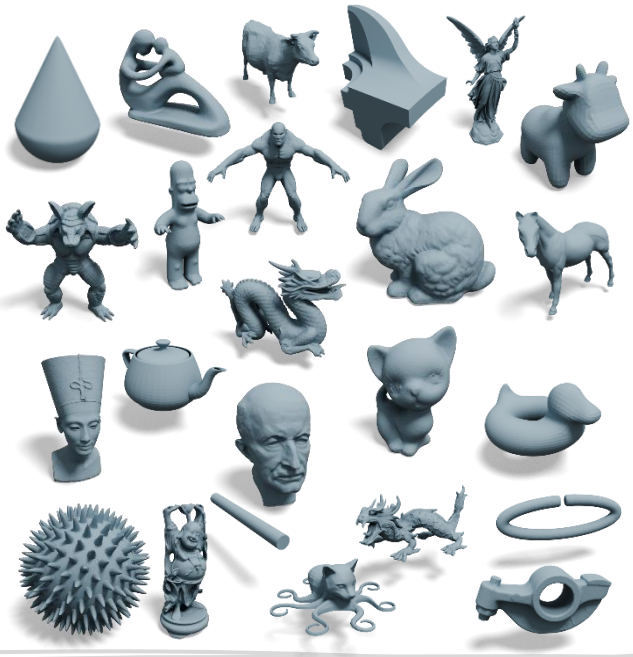


Performance - Poisson

- Best total time (if not tiny meshes)
- Lowest number of iterations needed

From 22k to 2.8M vertices

Model	#Vert N_1	Ours				Shi06				AMG-RS				AMG-SA				Direct (CHOLMOD)		
		Build	Solve	Total	#It	Build	Solve	Total	#It	Build	Solve	Total	#It	Build	Solve	Total	#It	Fact.	Subst.	Total
										Poisson Problems										
Fandisk	22k	0.09	0.12	0.21	7	0.06	0.19	0.25	10	0.05	0.22	0.27	49	0.07	0.11	**0.18**	39	0.17	0.00	0.18
Homer	33k	0.18	0.18	0.36	8	0.10	0.34	0.43	13	0.08	0.31	0.39	46	0.09	0.17	**0.26**	41	0.27	0.01	0.27
Teapot	52k	0.29	0.48	0.76	7	0.16	0.46	0.62	11	0.14	0.65	0.79	60	0.13	0.38	**0.51**	46	0.61	0.02	0.63
Horse	74k	0.46	0.39	0.84	9	0.22	0.64	0.86	14	0.20	1.01	1.20	62	0.19	0.60	**0.80**	48	0.78	0.02	0.80
Spike	102k	0.57	0.47	**1.04**	6	0.32	1.07	1.39	13	0.31	1.45	1.76	66	0.28	1.03	1.31	61	1.41	0.05	1.46
Max Planck	157k	0.76	0.80	**1.56**	8	0.56	1.89	2.44	14	0.48	4.15	4.63	116	0.64	1.71	2.35	58	2.26	0.07	2.32
Ring	208k	1.08	0.99	**2.07**	6	0.64	1.87	2.51	16	0.64	2.71	3.35	49	0.96	1.82	2.78	43	2.46	0.06	2.52
Rocker Arm	262k	0.99	1.41	**2.40**	8	0.83	2.45	3.28	16	0.84	4.60	5.44	73	1.12	3.04	4.16	58	3.35	0.09	3.44
Bob	309k	1.44	1.81	**3.24**	7	1.22	3.12	4.34	15	1.05	8.03	9.08	106	1.37	4.25	5.61	67	4.70	0.13	4.83
Cylinder	310k	2.45	1.87	**4.32**	6	1.09	3.38	4.47	17	1.27	7.04	8.31	87	1.65	4.73	6.38	64	4.75	0.14	4.88
Bunny	411k	2.12	2.71	**4.83**	7	1.28	4.43	5.72	14	1.48	14.63	16.11	142	1.95	5.69	7.64	60	7.74	0.20	7.94
Cube 80^3	512k	5.13	4.83	9.96	8	3.36	6.94	10.30	17	5.07	8.70	13.78	17	2.44	6.96	**9.40**	57	11.04	0.29	11.33
Armadillo	519k	4.35	4.08	**8.43**	11	3.08	6.20	9.28	16	2.60	30.09	32.69	192	2.80	10.12	12.92	71	8.87	0.21	9.08
Nefertiti	617k	4.32	5.10	**9.42**	10	2.54	7.59	10.13	15	2.41	41.30	43.70	233	3.16	11.48	14.64	67	13.78	0.35	14.13
Spot	709k	6.02	5.52	11.54	8	2.47	8.95	**11.42**	16	2.87	41.72	44.59	205	3.67	13.40	17.07	69	18.57	0.48	19.05
Spike Ball	829k	7.66	7.01	**14.67**	9	3.47	11.35	14.81	17	3.08	*timeout*	*> 1min*	285	6.30	16.09	22.39	63	35.79	1.22	37.01
Beast	881k	7.06	7.61	**14.68**	10	5.11	10.83	15.95	17	6.47	45.96	52.43	161	6.44	19.04	25.49	73	*failed*	*failed*	*failed*
Cube 100^3	1.0M	3.39	14.06	**17.45**	8	4.57	15.18	19.75	19	16.34	22.00	38.34	18	8.48	17.85	26.33	50	115.79	0.96	116.76
Dragon	1.0M	9.58	9.26	**18.84**	12	6.55	12.68	19.22	17	7.96	51.89	59.85	160	10.73	21.33	32.06	72	*failed*	*failed*	*failed*
Happy Buddha	1.2M	6.88	10.83	**17.70**	13	5.11	17.29	22.41	19	5.76	*timeout*	*> 1min*	184	6.90	27.29	34.19	87	45.51	0.65	46.16
Octocat	1.5M	7.23	12.34	**19.56**	9	5.94	22.04	27.98	22	7.09	*timeout*	*> 1min*	136	8.59	36.41	45.00	89	51.82	1.60	53.42
Kitten	1.5M	8.53	12.73	**21.26**	8	6.26	22.17	28.43	17	7.53	*timeout*	*> 2min*	287	9.04	30.98	40.02	72	60.04	5.81	65.85
Lucy	1.9M	21.93	18.41	**40.34**	16	12.43	28.30	40.73	20	13.45	185.55	199.00	260	19.98	38.41	58.38	89	52.14	1.38	53.51
Cow	2.0M	23.58	19.83	**43.41**	10	14.51	53.14	67.64	19	15.38	*timeout*	*> 4min*	237	23.12	87.46	110.58	87	*failed*	*failed*	*failed*
XYZ Dragon	2.4M	14.19	25.75	**39.94**	14	10.50	39.65	50.15	21	12.96	*timeout*	*> 2min*	153	15.68	69.47	85.15	98	*failed*	*failed*	*failed*
Fertility	2.8M	10.00	24.02	**34.02**	9	10.67	41.49	52.16	19	14.27	*timeout*	*> 2min*	146	17.47	94.11	111.59	86	*failed*	*failed*	*failed*

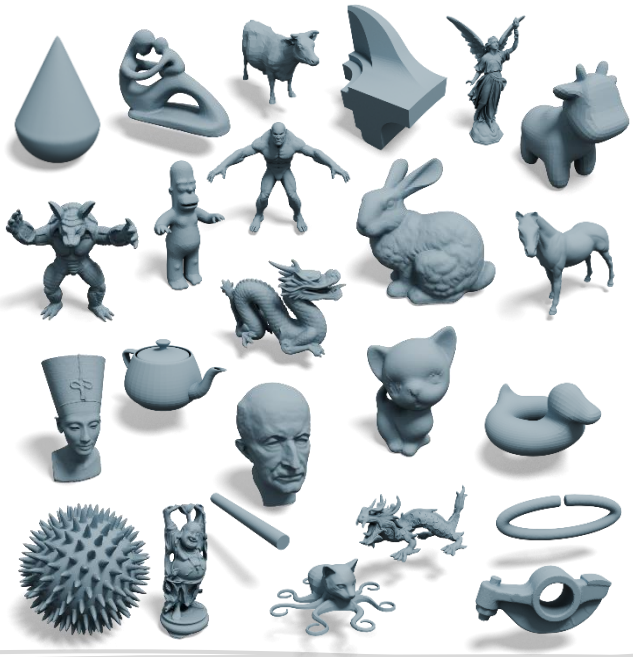


Performance - Biharmonic

- Best total time (if not tiny meshes)
- Lowest number of iterations needed

From 22k to 2.8M vertices

Model	#Vert N_1	Ours				Shi06				AMG-RS				AMG-SA				Direct (CHOLMOD)		
		Build	Solve	Total	#It	Build	Solve	Total	#It	Build	Solve	Total	#It	Build	Solve	Total	#It	Fact.	Subst.	Total
Biharmonic Problems																				
Fandisk	22k	0.12	0.32	**0.44**	24	0.08	0.67	0.76	52	0.06	2.43	2.48	290	0.06	1.18	1.24	156	0.52	0.01	0.53
Homer	33k	0.23	0.63	**0.87**	20	0.14	1.91	2.05	94	0.08	19.27	19.35	911	0.09	6.48	6.57	369	1.39	0.03	1.42
Teapot	52k	0.33	1.30	**1.63**	39	0.16	3.30	3.46	99	0.13	14.99	15.13	613	0.13	6.33	6.46	312	1.63	0.05	1.68
Horse	74k	0.45	1.49	**1.94**	26	0.25	5.70	5.95	133	0.20	58.91	59.11	1803	0.19	13.99	14.18	518	2.05	0.05	2.11
Spike	102k	0.66	3.34	**4.00**	47	0.31	15.36	15.67	236	0.29	*timeout*	*> 1min*	1156	0.27	33.29	33.56	778	4.29	0.11	4.40
Max Planck	157k	1.00	4.91	**5.91**	44	0.51	26.74	27.25	264	0.47	*timeout*	*> 1min*	721	0.61	*timeout*	*> 1min*	900	7.24	0.17	7.42
Ring	208k	1.17	4.68	**5.85**	26	0.60	15.01	15.61	101	0.63	*timeout*	*> 1min*	568	0.84	34.22	35.05	386	6.53	0.15	6.68
Rocker Arm	262k	1.47	5.76	**7.23**	24	1.64	27.48	29.12	153	0.83	*timeout*	*> 1min*	442	1.13	*timeout*	*> 1min*	539	9.41	0.22	9.63
Bob	309k	2.51	10.30	**12.81**	42	1.01	51.16	52.17	241	1.03	*timeout*	*> 1min*	344	1.42	*timeout*	*> 1min*	415	14.37	0.33	14.69
Cylinder	310k	1.96	9.81	**11.77**	38	1.08	41.47	42.56	193	1.07	*timeout*	*> 1min*	346	1.38	*timeout*	*> 1min*	422	12.81	0.30	13.11
Bunny	411k	2.22	18.85	**21.07**	56	1.28	99.69	100.97	329	1.47	*timeout*	*> 2min*	472	1.94	*timeout*	*> 2min*	553	30.82	1.39	32.21
Cube 80^3	512k	2.64	26.93	**29.57**	80	1.65	78.27	79.92	227	4.63	35.38	40.01	**51**	2.25	*timeout*	*> 2min*	507	40.13	0.97	41.10
Armadillo	519k	2.81	19.38	**22.20**	39	1.75	118.48	120.23	302	1.88	*timeout*	*> 2min*	358	2.53	*timeout*	*> 2min*	410	31.23	0.88	32.11
Nefertiti	617k	2.81	25.82	**28.62**	43	2.22	*timeout*	*> 2min*	266	2.39	*timeout*	*> 2min*	290	3.12	*timeout*	*> 2min*	326	56.01	7.24	63.25
Spot	709k	3.14	32.48	**35.61**	48	2.61	*timeout*	*> 2min*	230	2.87	*timeout*	*> 2min*	250	3.67	*timeout*	*> 2min*	283	*failed*	*failed*	*failed*
Spike Ball	829k	8.93	60.10	**69.03**	61	4.45	*timeout*	*> 4min*	353	4.93	*timeout*	*> 4min*	456	6.80	*timeout*	*> 4min*	303	*failed*	*failed*	*failed*
Beast	881k	6.30	42.16	**48.46**	51	4.94	*timeout*	*> 4min*	216	3.50	*timeout*	*> 4min*	236	3.23	*timeout*	*> 4min*	267	*failed*	*failed*	*failed*
Cube 100^3	1.0M	6.07	63.66	**69.72**	90	3.42	242.62	246.03	335	10.75	96.51	107.26	**69**	4.86	*timeout*	*> 4min*	484	*failed*	*failed*	*failed*
Dragon	1.0M	10.06	45.13	**55.19**	31	5.93	*timeout*	*> 4min*	193	6.60	*timeout*	*> 4min*	239	8.84	*timeout*	*> 4min*	313	*failed*	*failed*	*failed*
Happy Buddha	1.2M	11.39	72.70	**84.09**	44	8.09	*timeout*	*> 6min*	239	8.56	*timeout*	*> 6min*	264	11.52	*timeout*	*> 6min*	298	*failed*	*failed*	*failed*
Octocat	1.5M	7.98	53.31	**61.29**	27	5.97	*timeout*	*> 4min*	207	7.05	*timeout*	*> 4min*	224	8.74	*timeout*	*> 4min*	257	*failed*	*failed*	*failed*
Kitten	1.5M	8.47	75.77	**84.23**	46	6.18	*timeout*	*> 6min*	288	7.36	*timeout*	*> 6min*	322	8.96	*timeout*	*> 6min*	369	*failed*	*failed*	*failed*
Lucy	1.9M	11.84	89.17	**101.01**	43	8.02	*timeout*	*> 6min*	228	9.45	*timeout*	*> 6min*	255	11.29	*timeout*	*> 6min*	294	*failed*	*failed*	*failed*
Cow	2.0M	15.58	144.04	**159.62**	46	8.41	*timeout*	*> 8min*	140	16.38	*timeout*	*> 8min*	220	25.84	*timeout*	*> 8min*	342	*failed*	*failed*	*failed*
XYZ Dragon	2.4M	13.86	121.74	**135.61**	34	12.45	*timeout*	*> 8min*	200	12.96	*timeout*	*> 8min*	214	15.75	*timeout*	*> 8min*	246	*failed*	*failed*	*failed*
Fertility	2.8M	14.14	150.77	**164.91**	36	13.15	*timeout*	*> 10min*	200	15.48	*timeout*	*> 10min*	217	19.17	*timeout*	*> 10min*	246	*failed*	*failed*	*failed*



GravoTet

A Fast Multigrid Hierarchy Construction for Tetrahedral Meshes

- Fast mesh hierarchy construction
- Prioritized fast disk sampling
- Boundary first approach
- Near Delaunay Barycenter weights
- Fast total build + solving times
- Good MG convergence

Project page &
Supplementary code
marcelpadilla.com

Thank you for your attention!

INTERACTIVE GEOMETRY LAB

Acks: Alexander von Humbold Feodor-Lynen fellowship, Dutch Research Council (NWO)

SMI, July 2026

TU Delft
ETH zürich

Bonus Slides

That did not fit into the 15 min talk

Sparsity

Examples

Mesh	Method	#Vert N_l	#It Poisson	#It Biharm.	$\frac{\text{nnz}}{\text{row}}$ (avg/max)
Cylinder	Ours	310k; 38k; 3.9k; 428	**6**	**38**	3.50/4
	Shi06	310k; 82k; 17k; 3.5k; 738; 173	17	193	3.17/10
	AMG-RS	310k; 71k; 13k; 2.1k; 342	87	+346	2.06/8
	AMG-SA	310k; 11k; 143	64	+422	4.31/10
Spike Ball	Ours	829k; 101k; 10k; 1.2k; 310	**9**	**61**	3.44/4
	Shi06	829k; 216k; 45k; 9.2k; 2.0k; 523; 167	17	+407	3.11/11
	AMG-RS	829k; 190k; 34k; 5.5k; 968; 205	+285	+447	1.98/10
	AMG-SA	829k; 31k; 469	63	+504	4.65/12
Lucy	Ours	1.9M; 225k; 23k; 2.5k; 307	**16**	**43**	3.49/4
	Shi06	1.9M; 488k; 101k; 20k; 4.2k; 943; 224	20	+228	3.10/18
	AMG-RS	1.9M; 425k; 77k; 12k; 2.0k; 428	260	+255	2.02/10
	AMG-SA	1.9M; 68k; 820; 18	89	+294	4.18/13
Fertility	Ours	2.8M; 333k; 33k; 3.4k; 392	**9**	**36**	3.53/4
	Shi06	2.8M; 726k; 145k; 28k; 5.7k; 1.2k; 260	19	+200	3.11/17
	AMG-RS	2.8M; 633k; 113k; 18k; 2.8k; 536; 152	+146	+217	1.95/9
	AMG-SA	2.8M; 99k; 1.0k; 14	86	+246	4.01/12

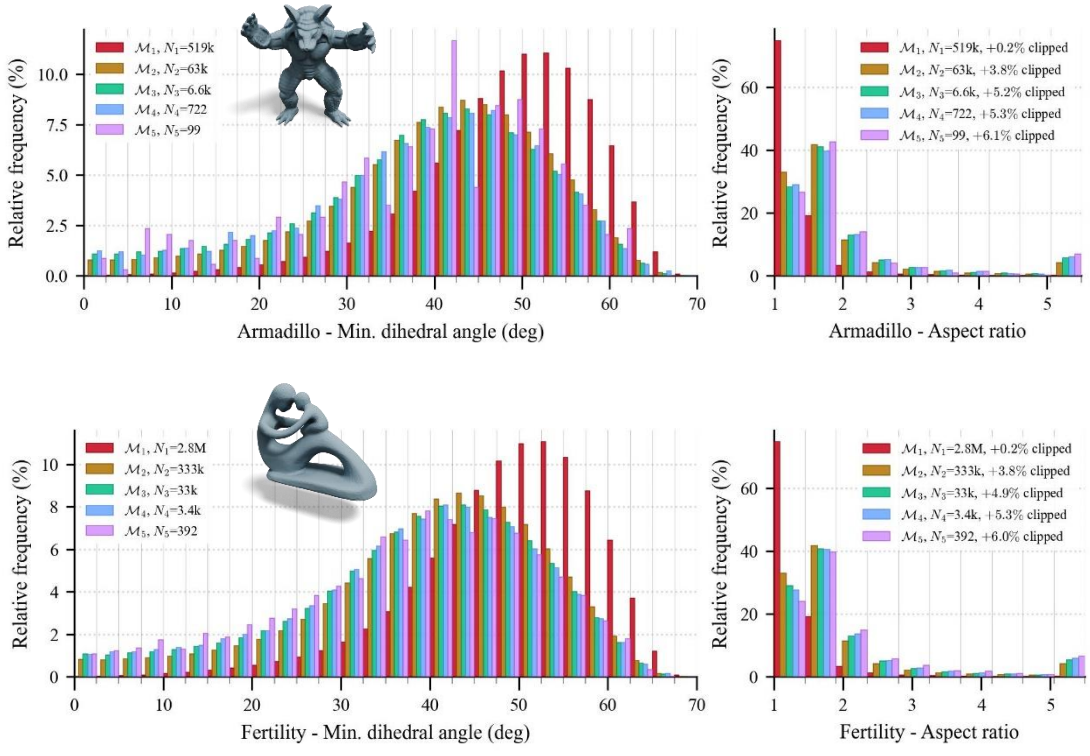
🌹 P_l has at most 4 non-zero entries per row.

Hiearchy Quality

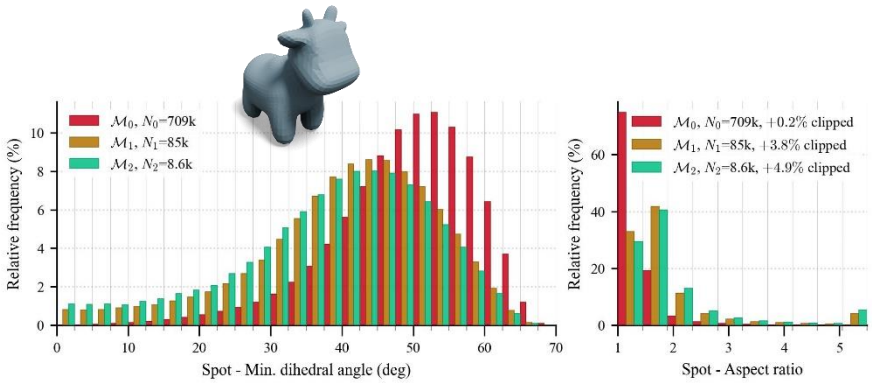
Low quality Tets are not a problem.

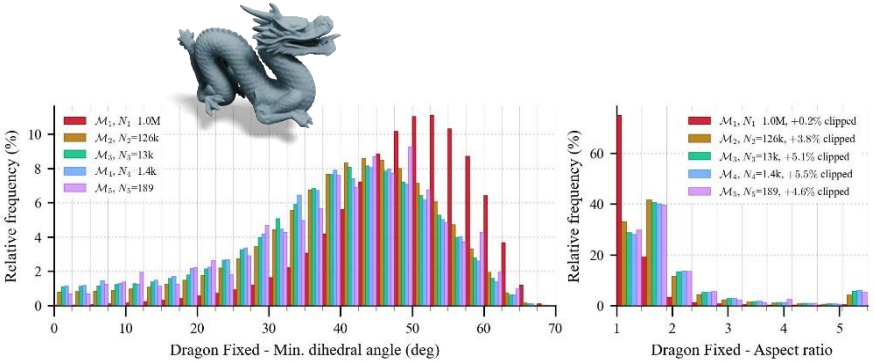
- Barycenter coordinates for propagation work fine! 👍

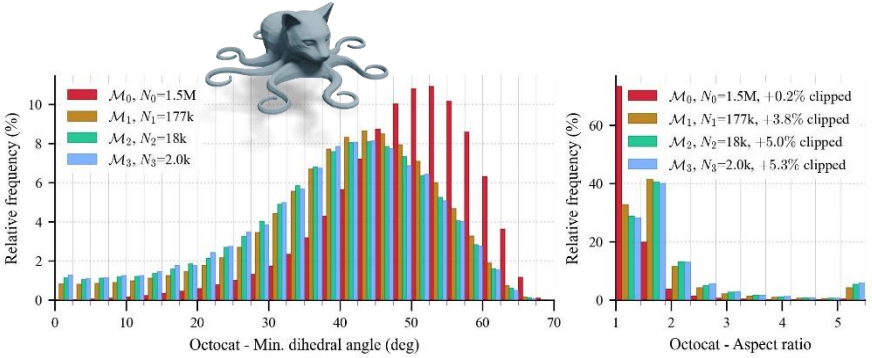
Red = input tet mesh

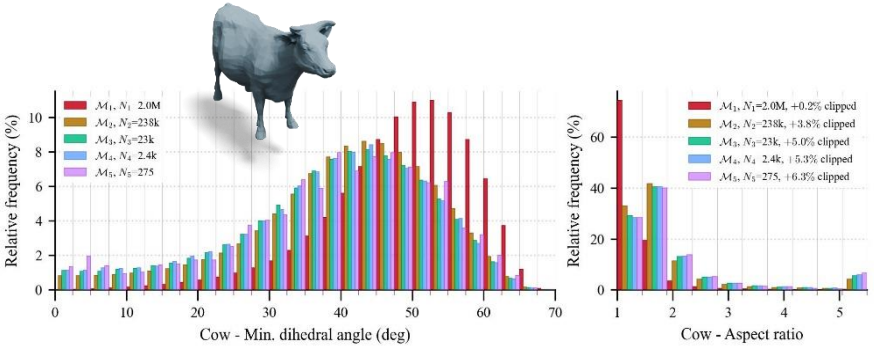


Hiearchy Quality

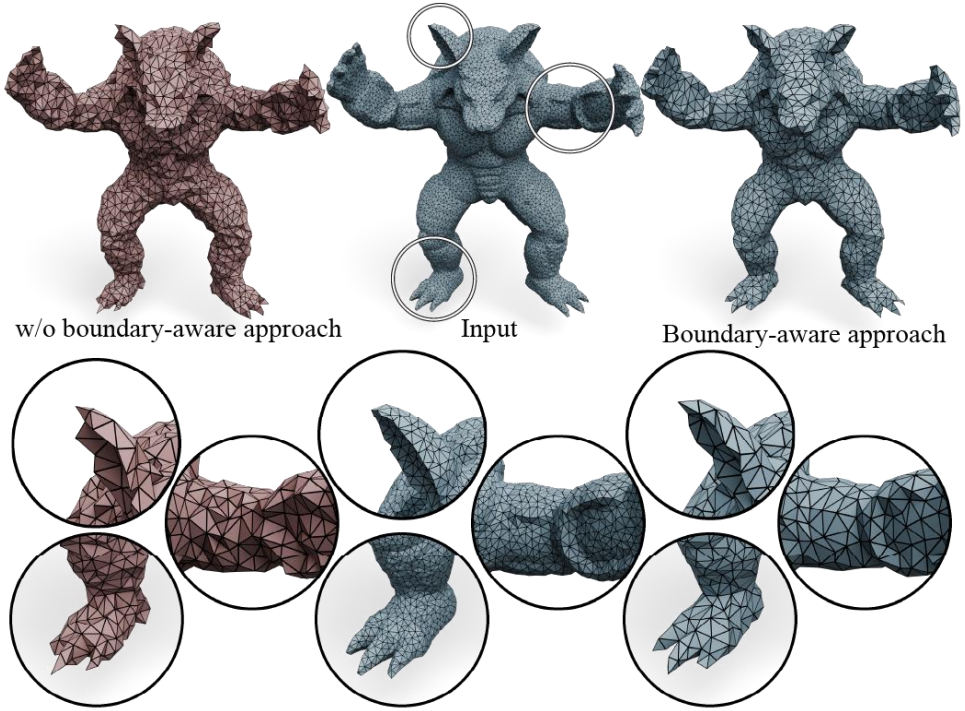








Ablation Study

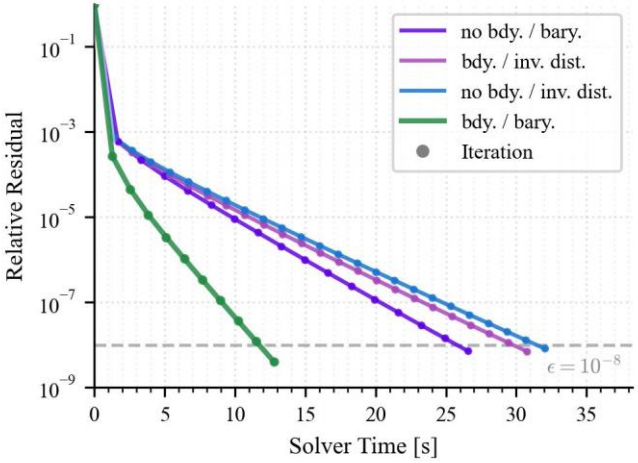


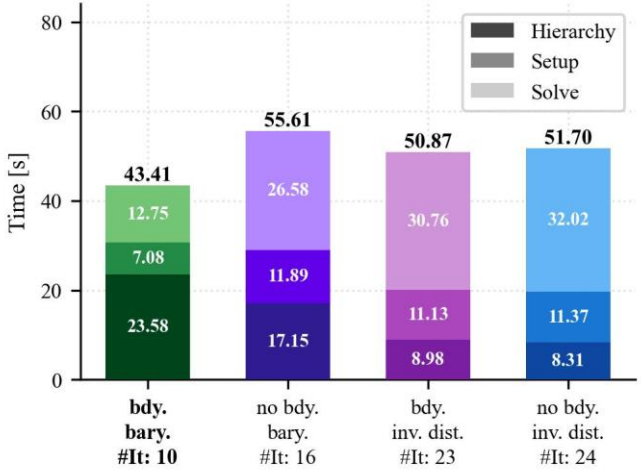
We vary:

- [Shi et. al. 2006]
  - Graph-based method
  - No-tetrahedralization (faster build)
  - Uses *inv. dist weights*
- Compare *without boundary handling*

Ablation Study

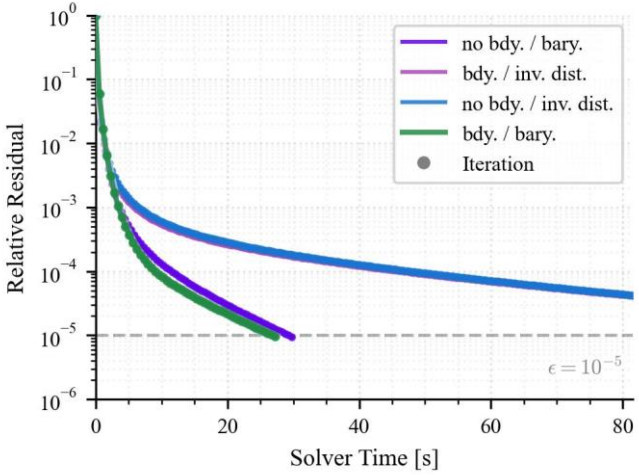
2M vertices. Poisson problem

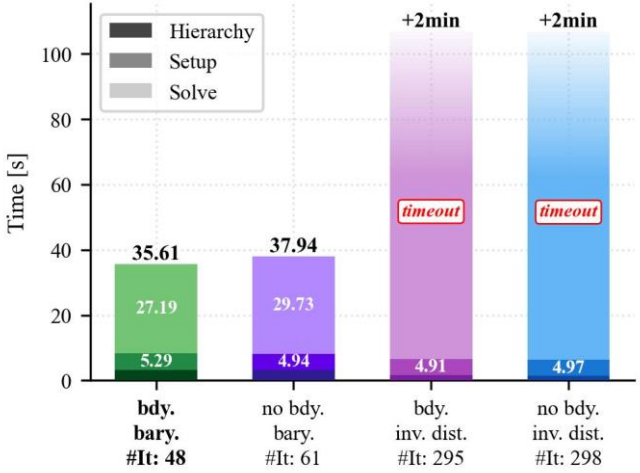




Ablation Study

709k vertices. Biharmonic problem





Multigrid methods

AMG

Algebraic Multigrid Methods

Use matrix connectivity only

- *Fast to build*
- Versatile
- *Slow to converge*

PyAMG
Algebraic Multigrid Solvers in Python

Based purely on matrix values

VS.

Only differ by 😳 prolongation matrix P_l

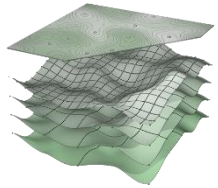
„Quality“ of P_l ✨ determines convergence

GMG

Geometric Multigrid Methods

Use Geometric Information

- *Slower to build*
- Less Versatile
- *Faster to converge*



Using mesh information. Distances, coordinates, mesh connectivity.

https://hpc.uni-wuppertal.de/en/research/topics/multigrid-methods/

Experiment Setups

Poisson problem

$$S\,x = M\,y$$

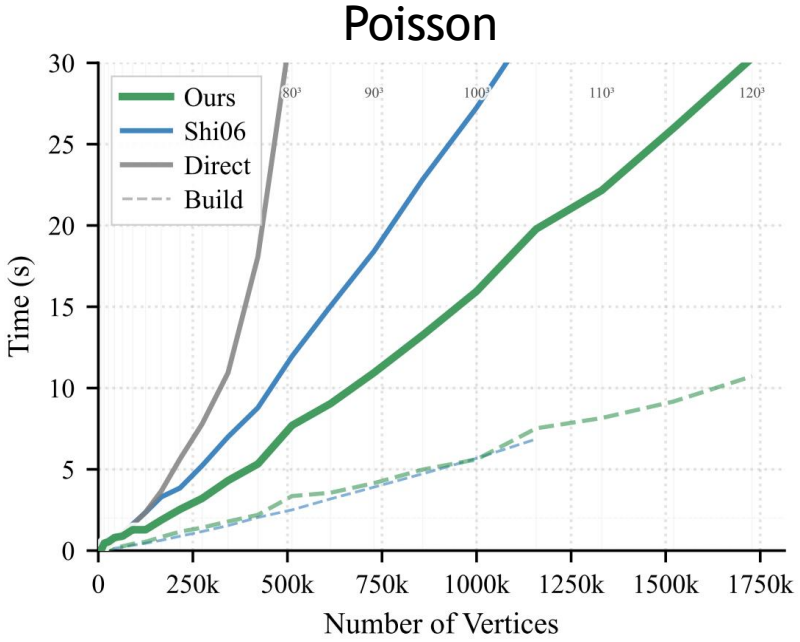
Biharmonic problem

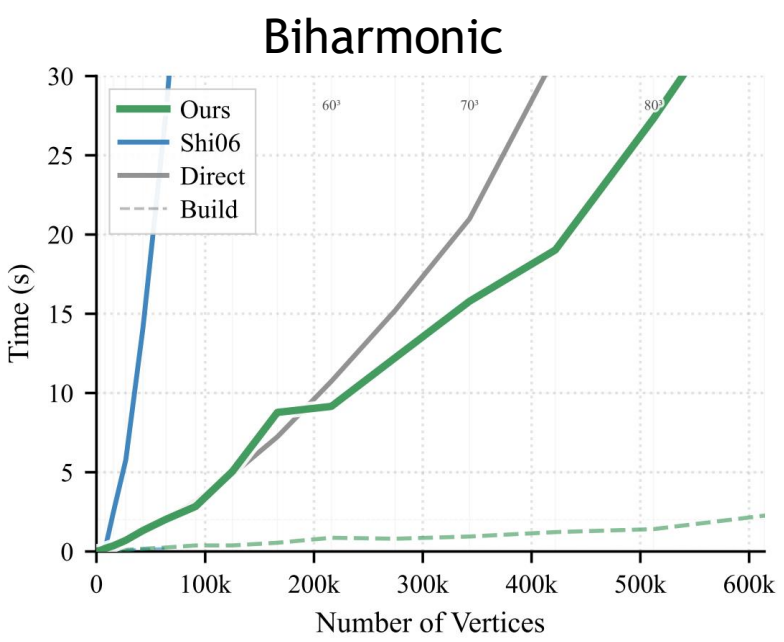
$$S\,M^{-1}\,S\,x = M\,y$$

Stopping criteria: $\|S\,x - M\,y\|_{M^{-1}} < \epsilon\,\|M\,y\|_{M^{-1}}$

Complexity

Cubes of increasing resoltions n^3

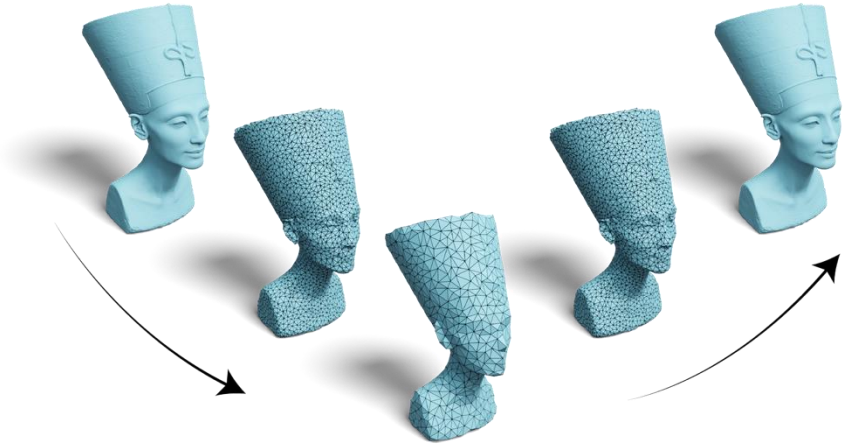




GravoTet

Most related paper ❤️

A Fast Geometric Multigrid Method for Curved Surfaces

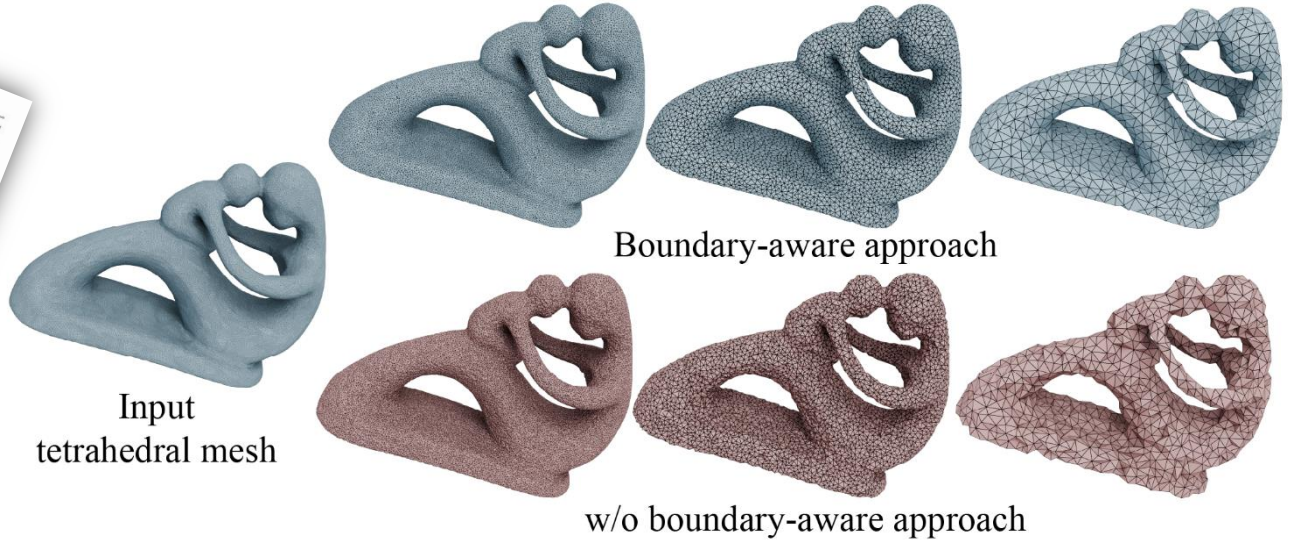


⚠ Fast disk sampling + Delaunay approximation, but ONLY on surfaces! ⚠

🧠 Transfer the logic to tetrahedra + careful boundary handeling 🧠

GravoTet

Most related paper ❤️ A Fast Geometric Multigrid Method for Curved Surfaces



🧠 Transfer the logic to tetrahedra + solve the emerging problems 🧠