Simplifying Unstructured Grids for Oceanographic Visualization

Ole Tytlandsvik

April 28, 2025

Oceanbox

- Tromso-based
- Interactive oceanographic simulations
- Oceanography as a Service
- Web-Based Geographic Information System (Web GIS)
- Digital twin of the coastal ocean

The Problem

- Large data sets
  - High resolution (millions of spatial points)
  - Multi-dimensional
  - Payloads of 20Mb+
  - Unresponsive web application
  - Increased bandwidth costs

The Solution: Lossy Compression

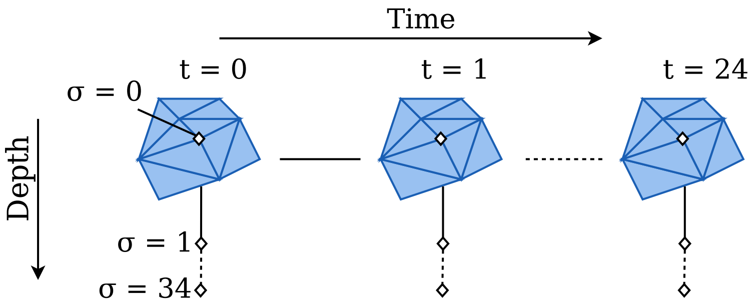
- Traditional approaches tricky
  - Accuracy of coordinates are important
  - Tiling/Multi-resolution not trivial with *unstructured grids*
  - Inflated data size should be smaller on the client

The Solution: Lossy Compression

Hybrid approach:

- Grid simplification
  - Remove vertices/nodes
  - Maintain visualization quality
  - **Angle bounded edge collapse**
- Floating-point compression
  - Compress one-dimensional vectors
  - Retain enough precision for visualization
  - **The ZFP compressor**

FVCOM grids



- Unstructured grid
  - Variable resolution
- Multiple dimensions
  - Coordinates, depth, time

Related Work

Rasterization-based approaches

- Good compression rates (90%)
- Requires *structured* grid layout

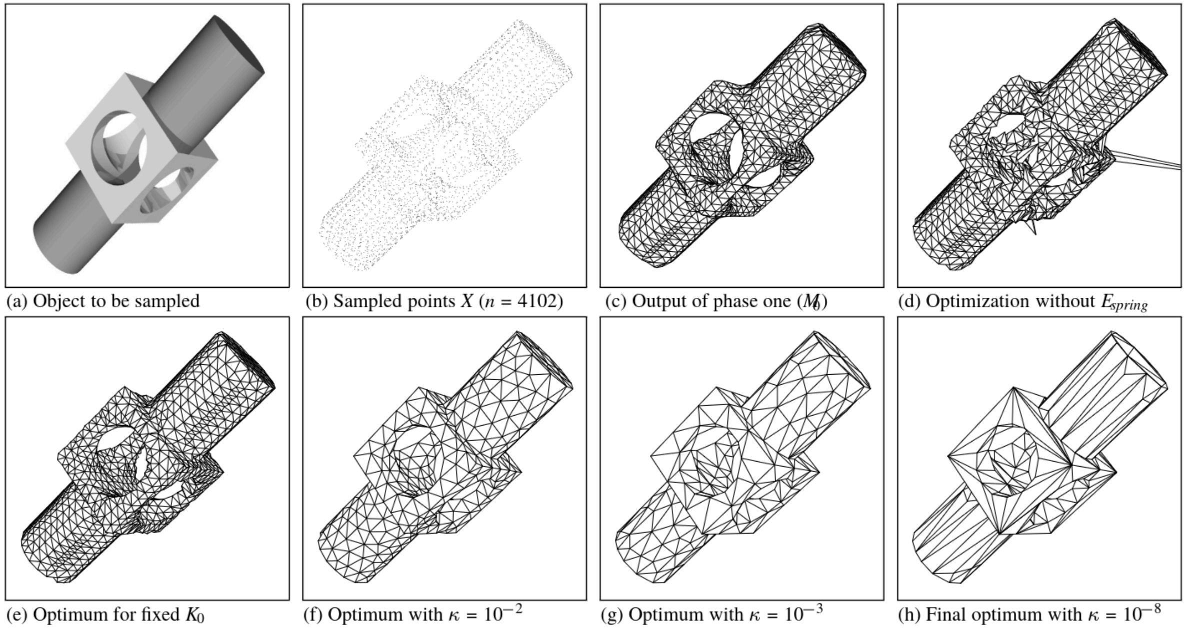
Tiling approaches

- Load only what you need
- Varying resolution
- Also requires structured grids

Mesh simplification

- More suited for unstructured grids
- Popular in literature: *3D mesh decimation*
- Not necessarily directly applicable...

Related Work: Mesh Simplification



(a) Object to be sampled (b) Sampled points X (n = 4102) (c) Output of phase one (M_0) (d) Optimization without E_{spring}

(e) Optimum for fixed K_0 (f) Optimum with $\kappa = 10^{-2}$ (g) Optimum with $\kappa = 10^{-3}$ (h) Final optimum with $\kappa = 10^{-8}$

- Approximate a surface
- Preserve topology
- Good reduction[1]

However

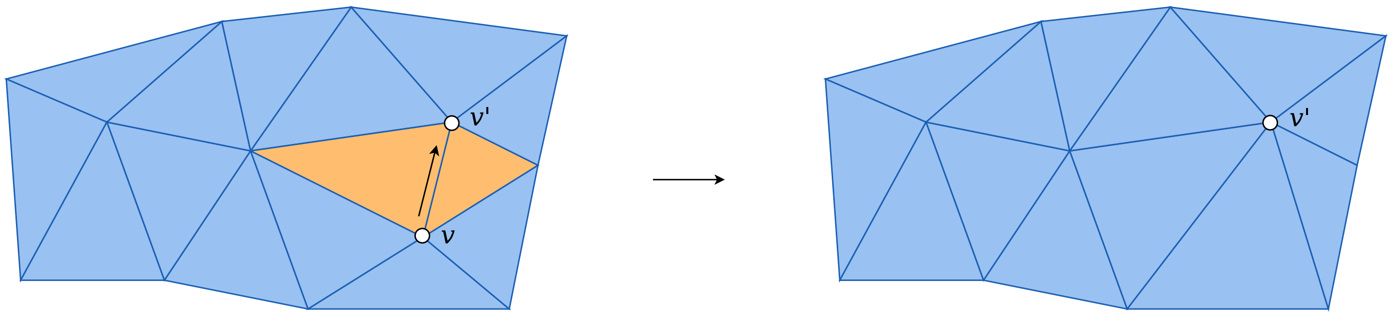
- Not ideal for 2D grids
- Need even resolution

[1]Figure from Hoppe et al. "Mesh Optimization", *ACM*, 1993, pp. 19-26

In the Capstone: Angle Bounded Half-edge Collapse

Adaptation from previous work.[1]

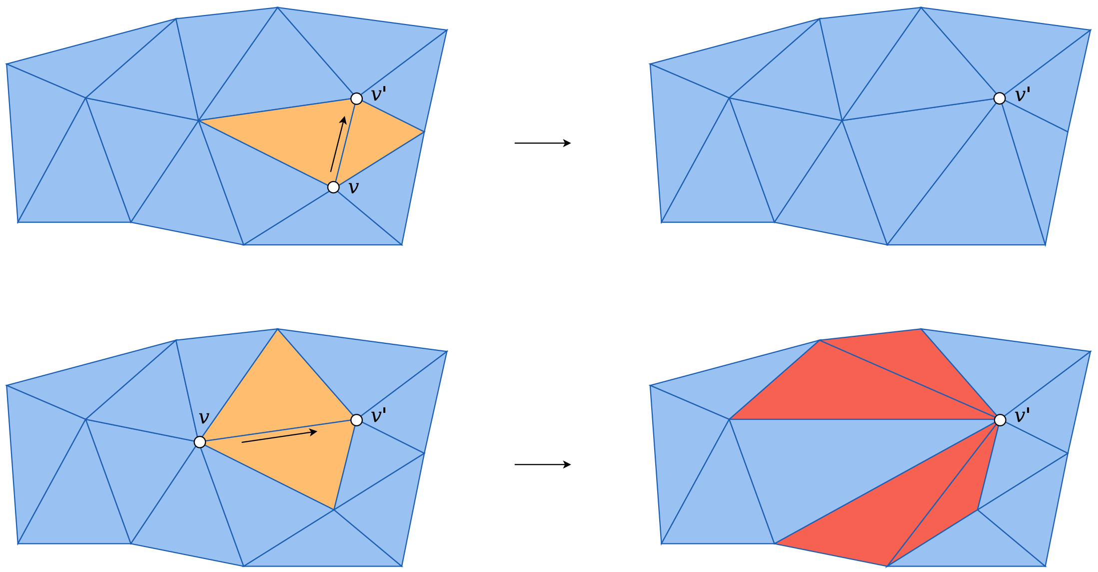
Half-edge collapse with a minimum angle criterion to inner angles



[1] Hinderink et al. "Angle-Bounded 2D Mesh Simplification." *Computer Aided Geometric Design*, vol 95, May 2022, p. 102085

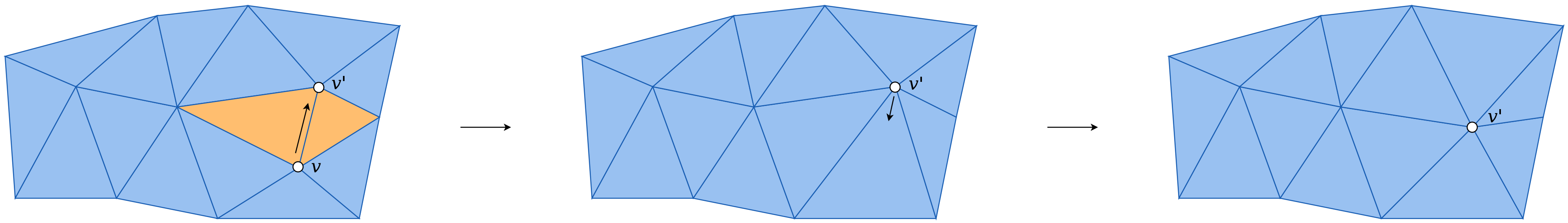
In the Capstone: Angle Bounded Half-edge Collapse

- We collapse v into v' by collapsing the half-edge $v \rightarrow v'$
- We define a strict angle bound θ
- We ensure inner angles θ_n respect $\theta_n > \theta$



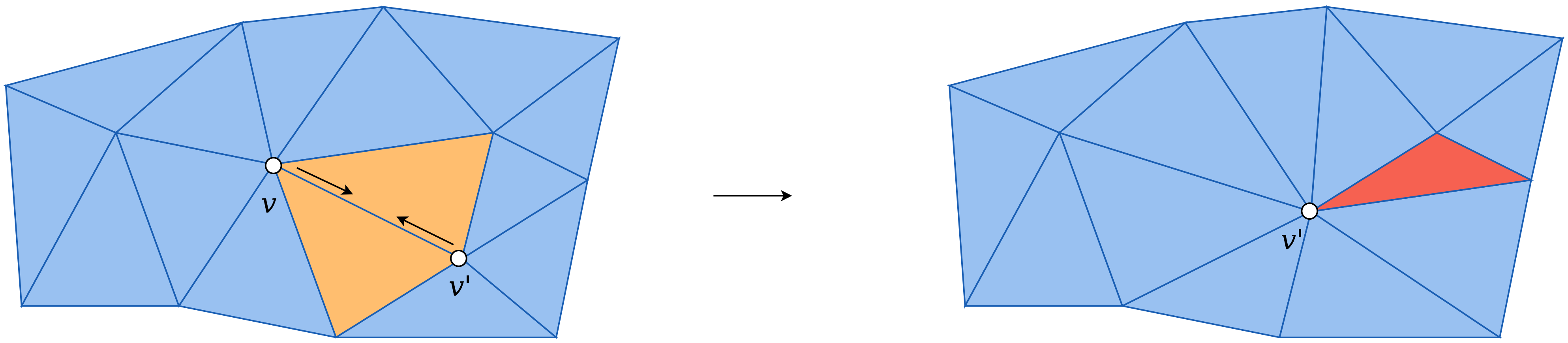
Improvement: Angle Bounded Edge Collapse

- Can be seen as a direct advancement of the half-edge collapse
- Simply use the average position of v and v'



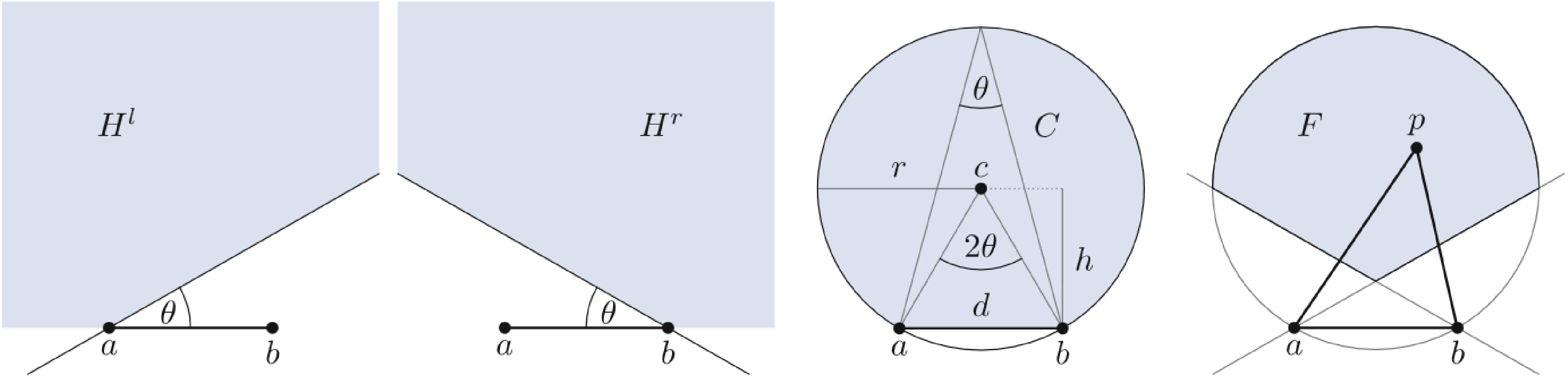
Improvement: Angle Bounded Edge Collapse

- Still enforce the same angle bound θ
- More collapses possible, quality better preserved



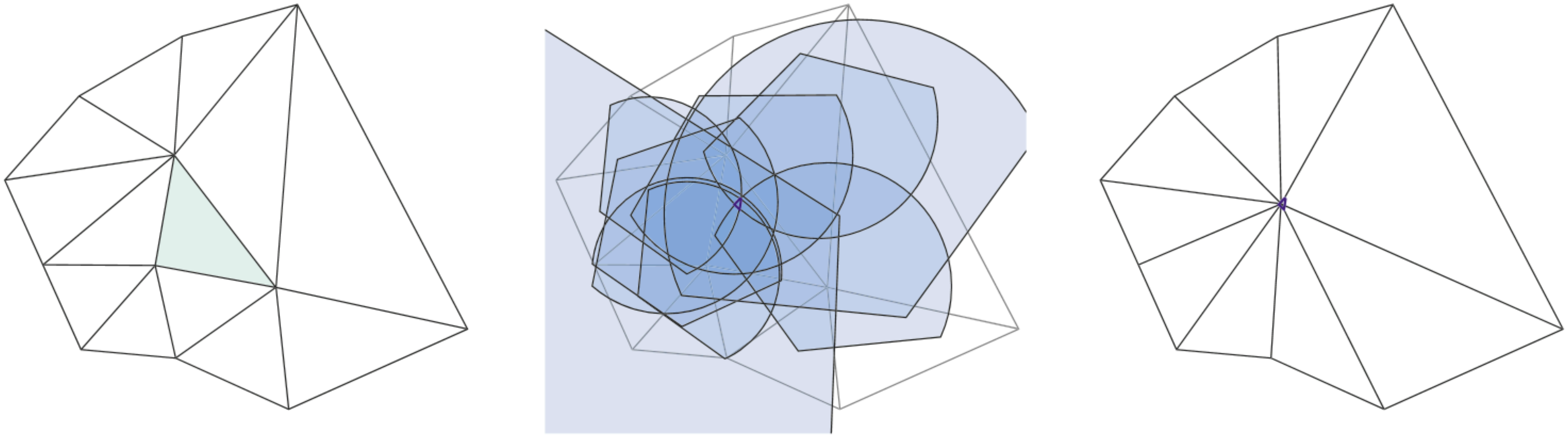
Merger Vertex Optimization

- Trivial optimization is using the centroid (what we just saw)
- The next step: *Kernel mean construction*



(a) Halfplane through *a*. (b) Halfplane through *b*. (c) Circle through *a* and *b*. (d) Common intersection.

Merger Vertex Optimization



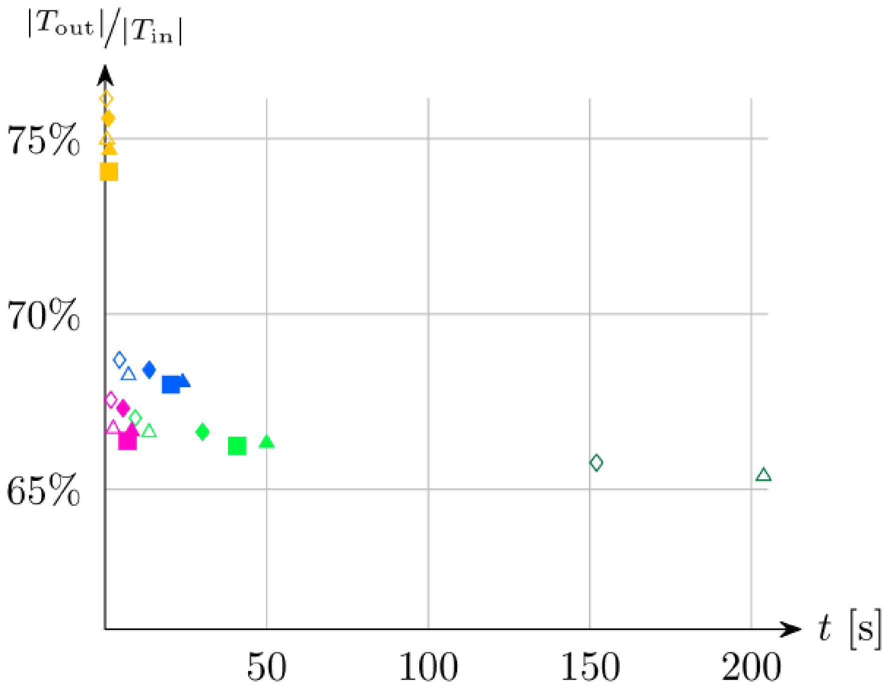
(a) Triangle and surround before the collapse. (b) All nine fans and their intersection K. (c) Possible configuration after the collapse.

Fig. 5. Angle-bounded kernel (bold blue), defined as the intersection of fans (light blue), when collapsing a triangle (green).

Merger Vertex Optimization

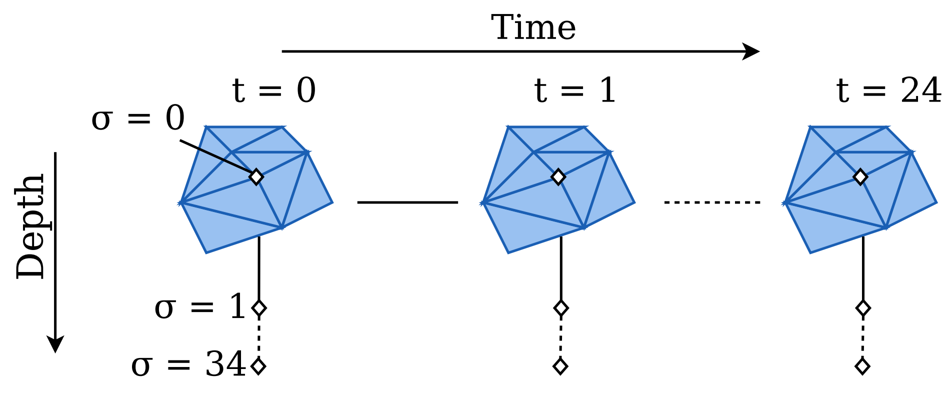
- Yields better results in the paper
- If a solution exists, collapse is guaranteed

Orange is centroid, blue is kernel mean construction →



Floating-point Compression

- Interactive visualizations:
  - Each frame is one value per node
  - One-dimensional array slices
  - Floating-point values (doubles)
  - **Challenge:** compress doubles on server, decompress on client



- ZFP is C++
  - Was able to compile to WASM!

Evaluation (so far)

Visualization similarity

- Inspection of raster images
  - Grid compression
  - Floating-point compression

Compression

- Grid geometry
- Floating-point slices
- Combined

Evaluation (so far)

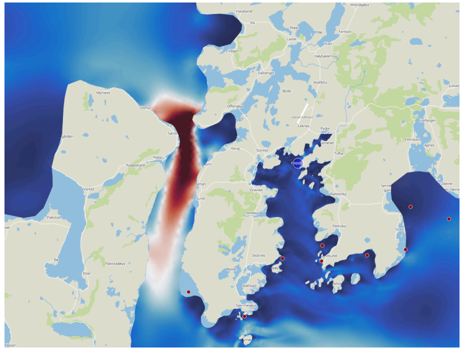
Grid: Buksnes waste (Nappstraumen in Lofoten)

Comparison

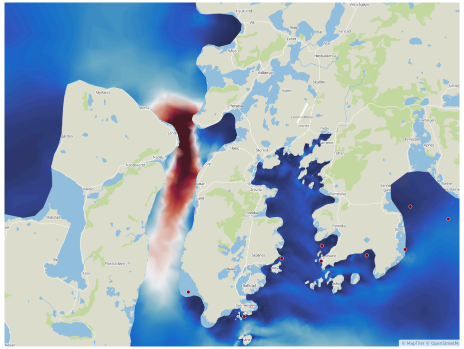
- Original grid
- Angle Bounded Half-edge Collapse
- Angle Bounded Edge Collapse
- Both with a range of values for θ

- *Kernel Mean Optimization not quite working yet*

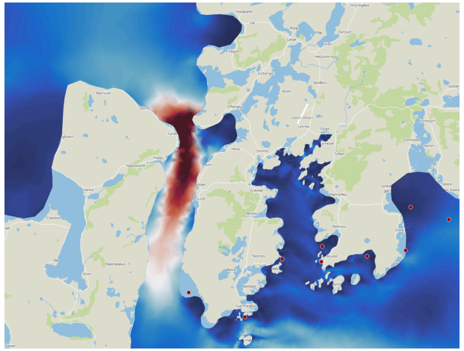
Visualization Similarity: Speed



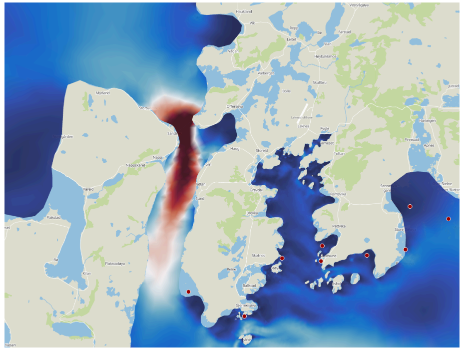
Original grid



Random reduction

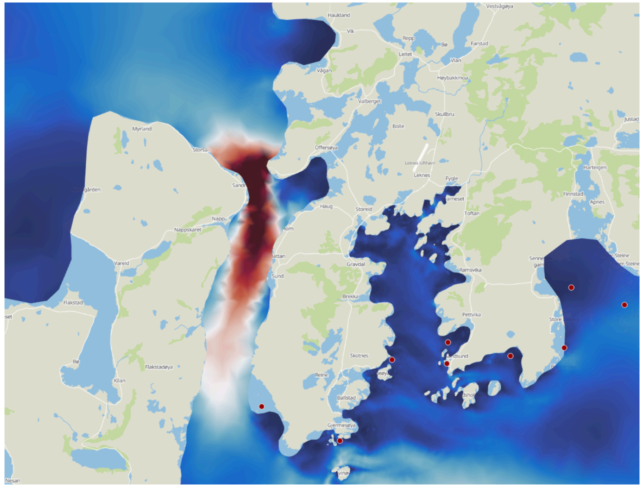


Half-edge, 28°

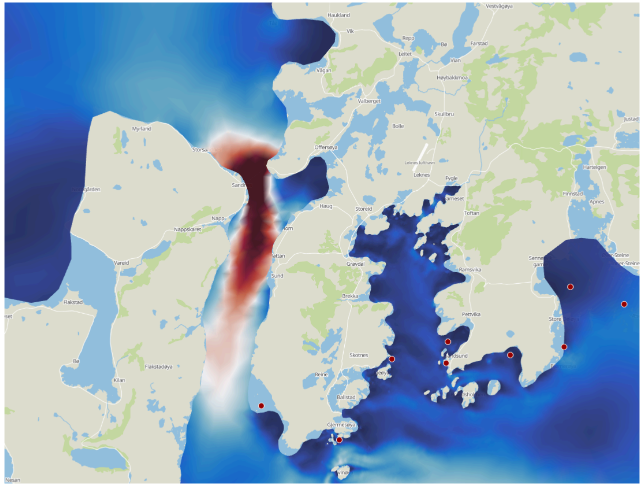


Full-edge, 28°

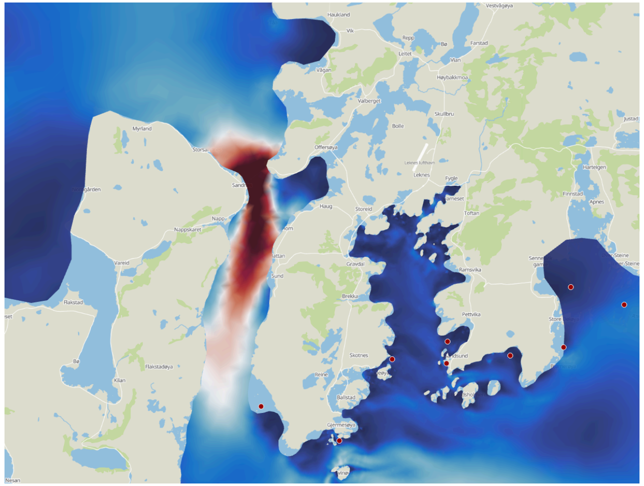
Visualization Similarity: Speed



Half-edge, 28°

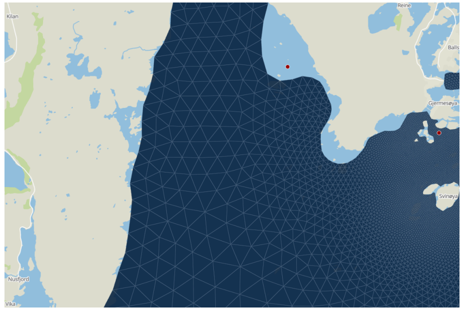


Full-edge, 28°

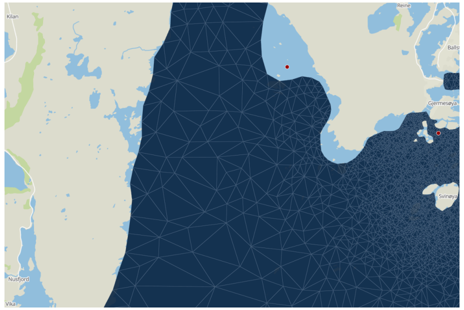


Full-edge, 40°

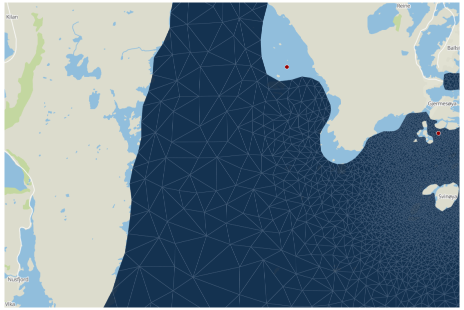
Visualization Similarity: Triangulation



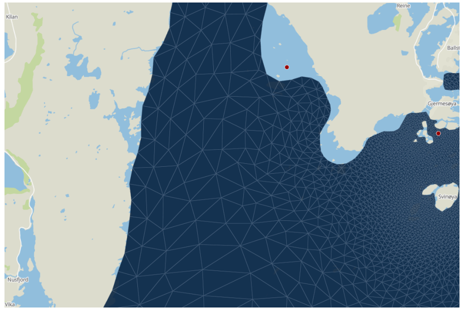
Original grid



Random reduction

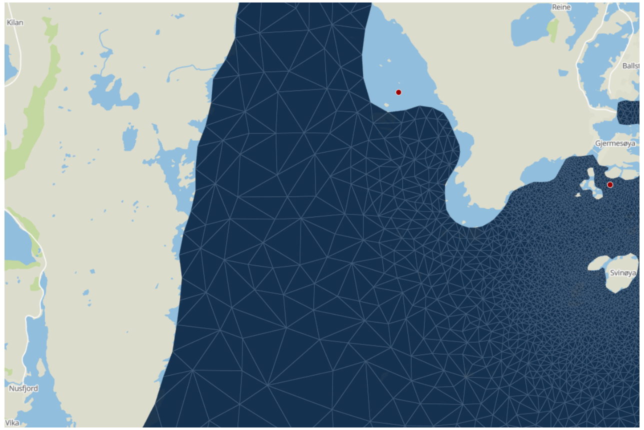


Half-edge, 28°

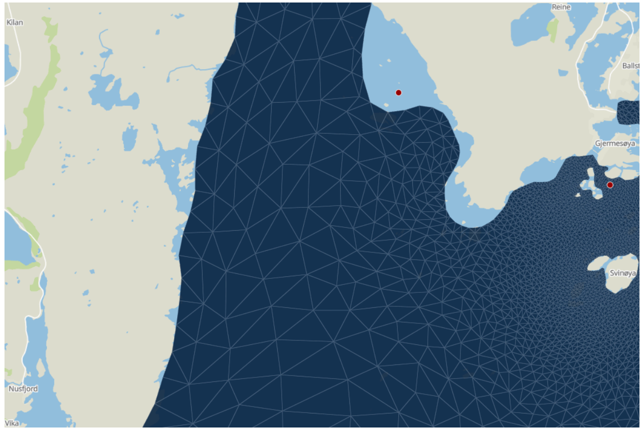


Full-edge, 28°

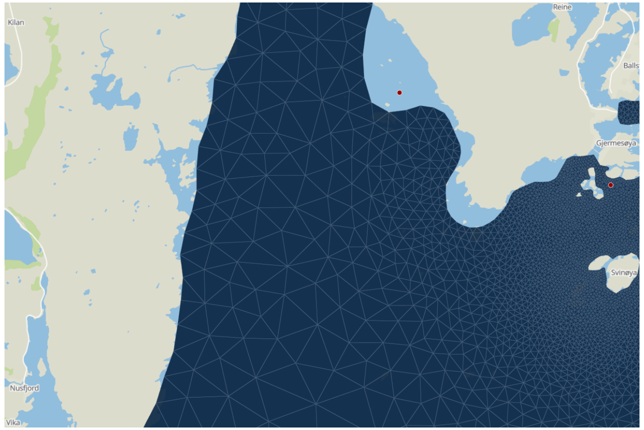
Visualization Similarity: Triangulation



Half-edge, 28°



Full-edge, 28°



Full-edge, 40°

Compression Ratio

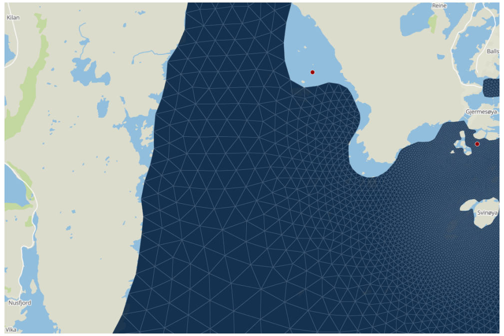
Angle Bound	Size / Compression Ratio			
	Half-edge		Full-edge	
	Nodes	Elements	Nodes	Elements
Full resolution	25 136	48 332	25 136	48 332
28°	1.71	1.76	1.80	1.86
30°	1.43	1.45	1.77	1.82
34°	1.09	1.09	1.74	1.79
40°	1.00	1.00	1.43	1.46

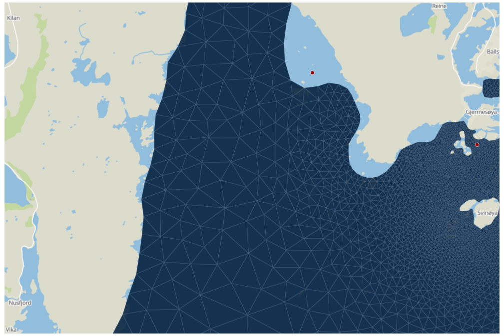
Compression Ratio

Angle Bound	Size / Compression Ratio			
	Half-edge		Full-edge	
	No ZFP	ZFP	No ZFP	ZFP
Full resolution	98KiB			
28°	1.71	5.76	1.80	6.12
30°	1.43	5.16	1.77	6.13
34°	1.09	3.92	1.74	5.76
40°	1.00	3.63	1.43	4.90

Summary

- **Grid Simplification** and **Floating-point Compression**
- Angle bound half-edge collapse vs edge collapse
  - Edge collapse significantly better
- Improved results so far
  - Better visualizations
  - 1.7x $\rightarrow$ 6x compression (array slices)





Remaining Work

- Working Kernel Mean Optimization
- Explore ZFP configuration
- Evaluation
  - Visualization results for ZFP
  - Pixelwise difference
  - Timings

Questions