Simplification and Streaming of GIS Terrain for Web Client

Web3D 2012

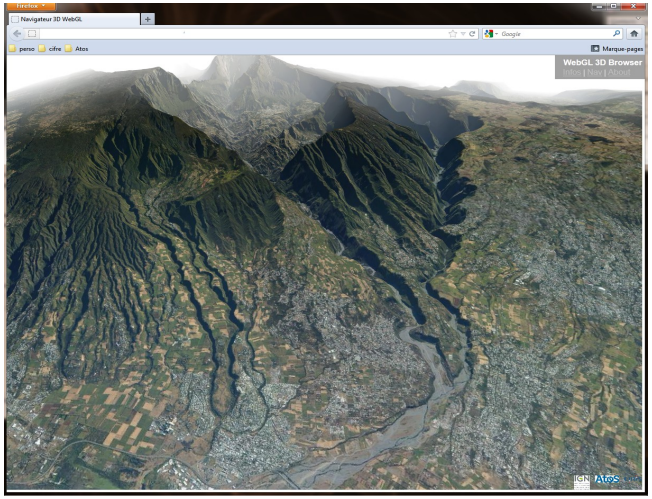
Geomod – Liris : Pierre-Marie Gandoin
Geomod – Liris : Raphaëlle Chaine
Atos Wordline : Aurélien Barbier-Accary

Fabien CELLIER
Université de Lyon
UMR CNRS 5205
Geomod

- ***Motivation***
- State of the Art
- Parallel Simplification
- Compression – Decompression
- Results and Conclusion

Motivation

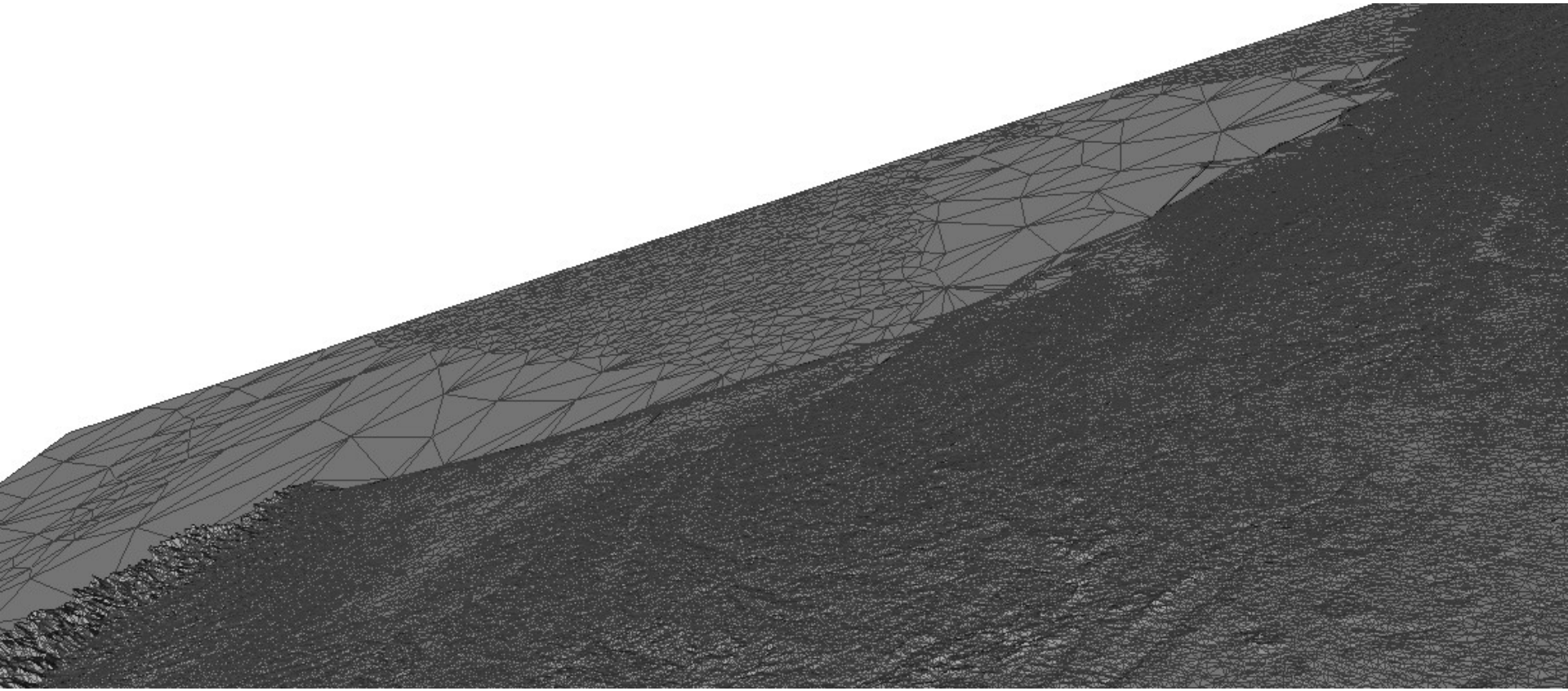
- **OpenscalesGL**, NokiaMaps... 3D GIS (geographical information system) comes to the browser.



- Possibility to reuse algorithms (with some refactoring).
- New problems : availibility, scalability and efficiency to address many devices (Desktop, tablet, smartphone).

Motivation

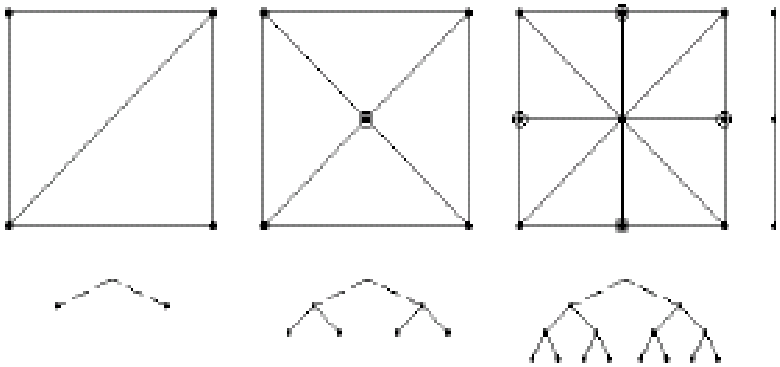
- ***Parallel*** (GPU), ***streaming***, surface meshes, compatible with out of core solution

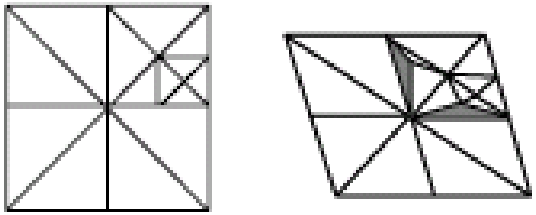


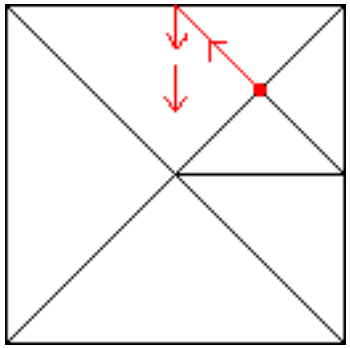
- Motivation
- ***State of the Art***
- Parallel Simplification
- Compression – Decompression
- Results and Conclusion

State of the art
Bdam (P. Cignoni et al.)

- Binary tree segmentation
  - Each node is a triangular patch
  - Used for localization
- Organisation of patches
  - ***Irregular triangulation*** in patches
  - Constrained vertices on borders which are shared by the patches
- Constraints in the tree to avoid « cracks »
- Out of core computation
- Limit
  - No streaming

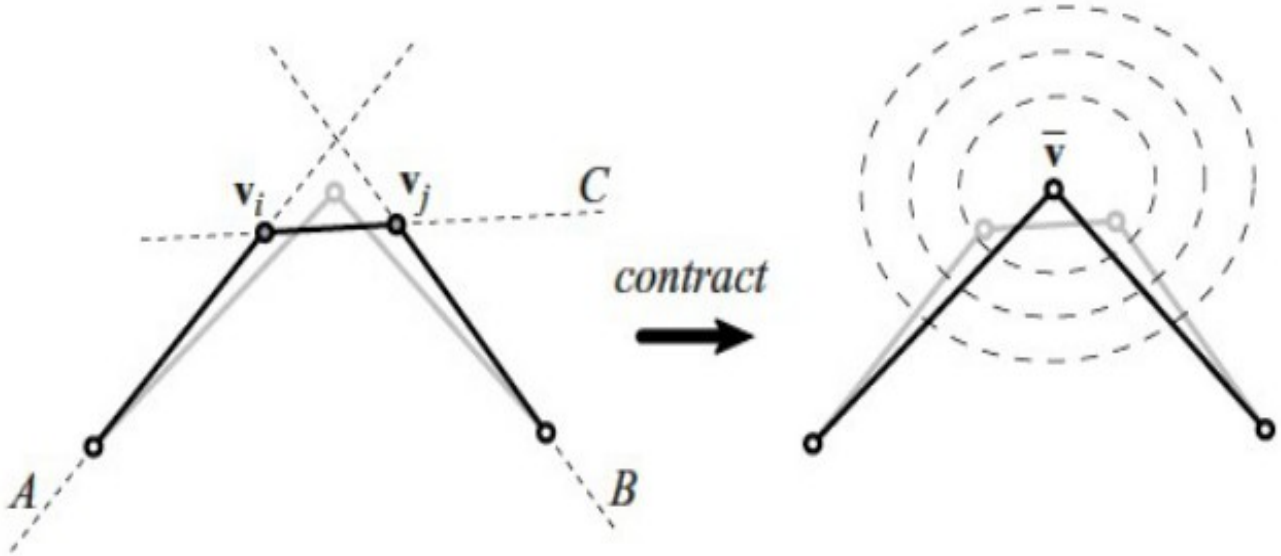






State of the art
QEM (Garland/Heckbert)

- Iterative simplification algorithm by edge collapse
- Metric used :
  - The introduced error should minimize the sum of squared distances to the planes
  - Represented by a 4x4 matrix Q per vertex
  - Q : keep track of the distance between new vertex and original planes
    → ***Enable parallelism***



State of the art

Progressive Meshes (Hoppe et al.)

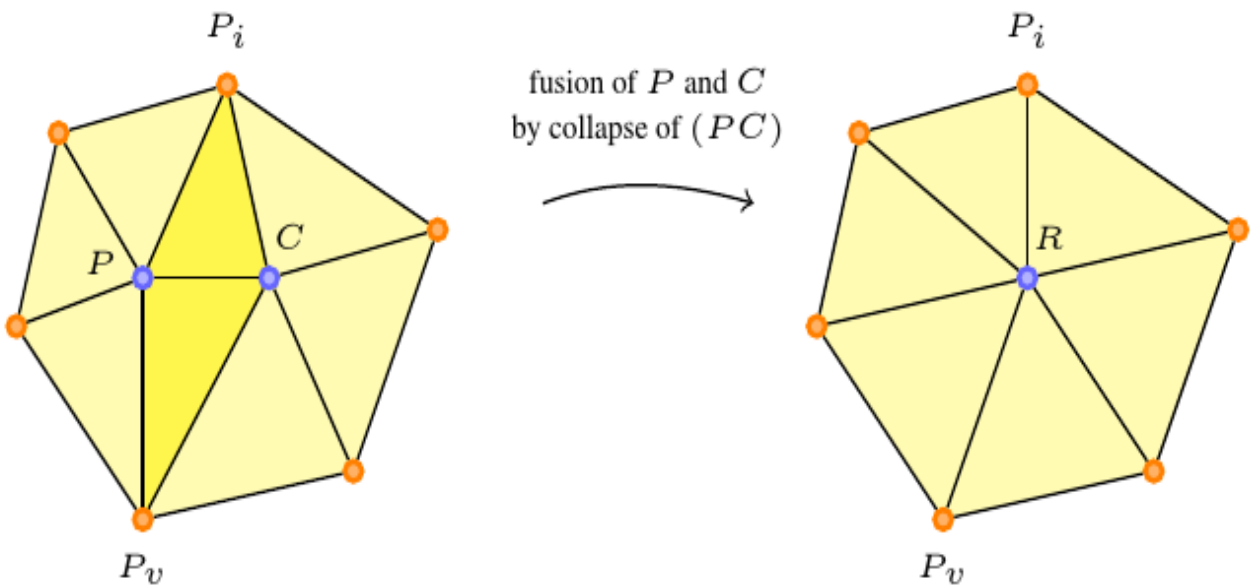
- Reverse edge compression ***iteratively***

- Why not doing parallelism on each tile ?
  → doesn't fit GPU architecture
  → iterative in patch
    → more patches

- Motivation
- State of the Art
- ***<u>Parallel Simplification</u>***
  - Our method
  - Consequences
- Compression – Decompression
- Results and Conclusion

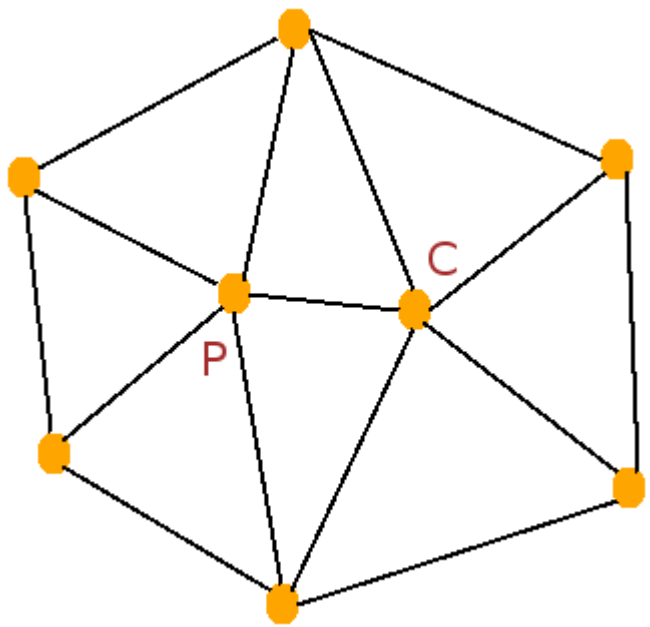
Parallel Simplification
Our method

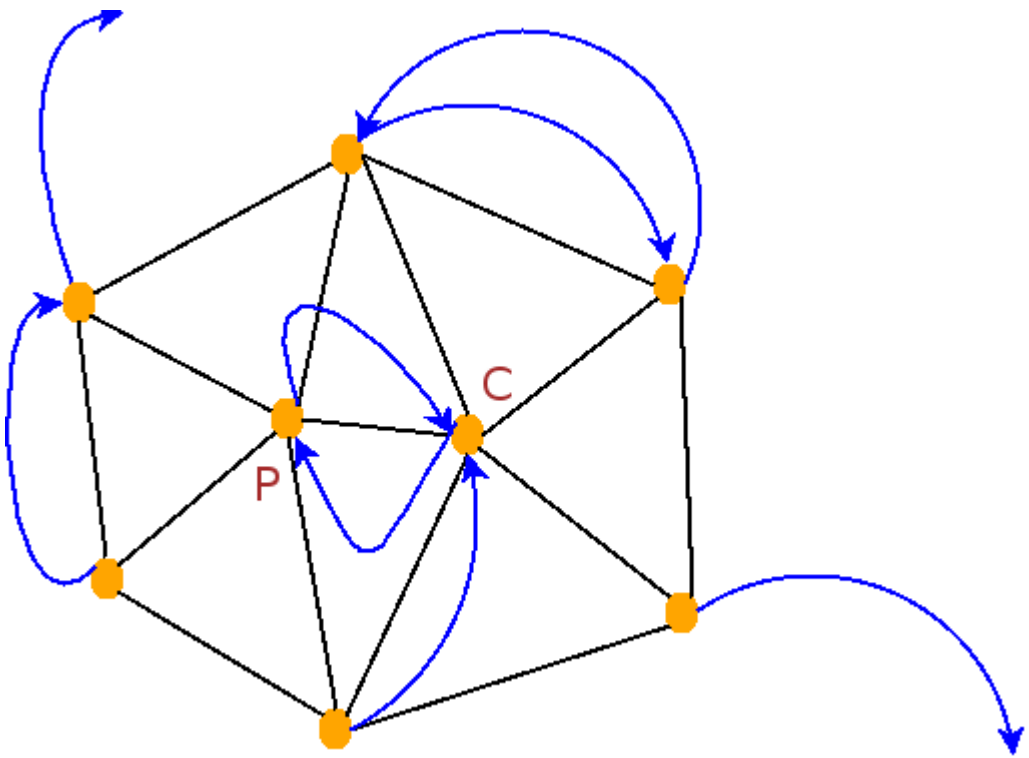
- Some concepts :
  - Edge collapse, vertex fusion/ split, 1 ring



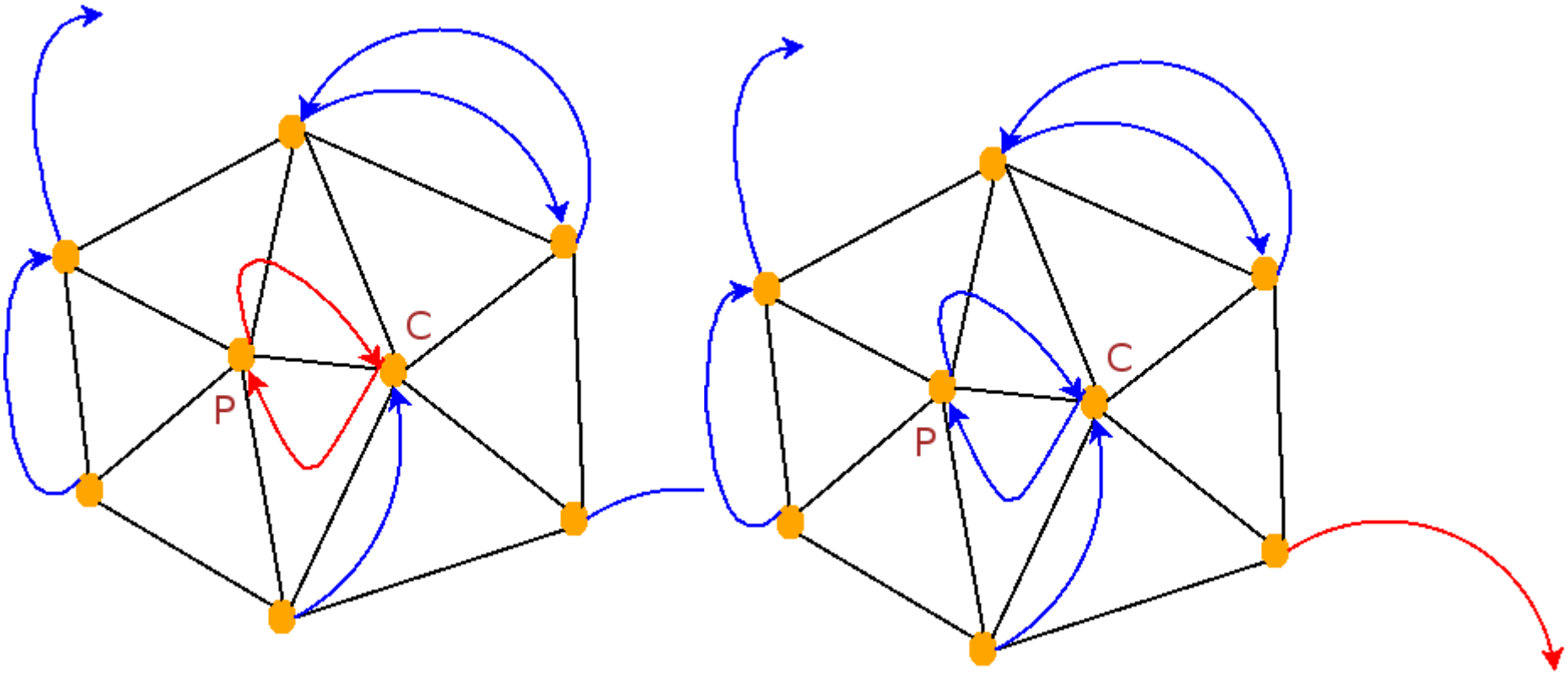
Parallel Simplification
Our method

- No more priority queue
  - Best candidate for fusion (per vertex)
  - Determined for each vertex in parallel
  - Priority is computed meanwhile





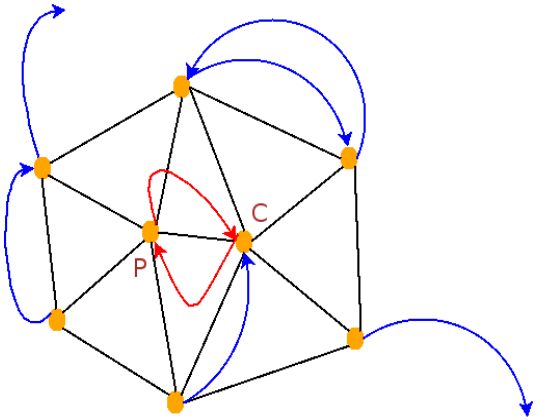
Parallel Simplification
Our method



Cancel

The worst case is incremental (only one fusion at a time)

Parallel Simplification
Our method



- The algorithm :
  - *For each* vertex in parallel : find candidate and compute the priority
  - *For each* vertex in parallel : perform the collapse if possible :
    - Store R instead of P
    - Update triangle vertices and the mesh connectivity

  - ***Parallel sort*** (needed to locate valid vertex and perform next parallel iteration)

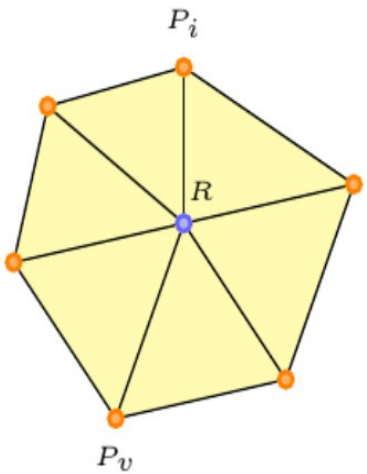
Parallel Simplification
Consequences

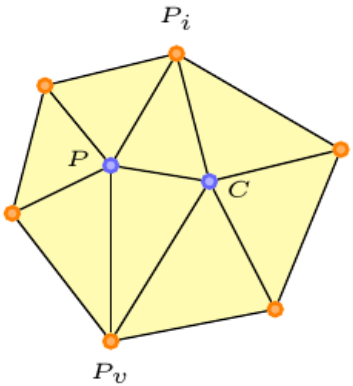
- Worst case is only iterative simplification
- Parallelism is on data (vertex and triangle)
- When P and C perform the fusion, 1 ring will be blocked (implicit lock)
  → easier to keep the mesh integrity
  → increase the result quality (minimal local error)

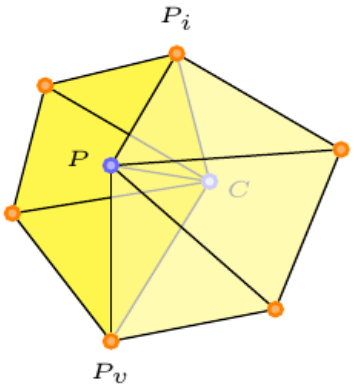
- Motivation
- State of the Art
- Parallel Simplification
- ***<u>Compression – Decompression</u>***
  - Requirement
  - Our method
- Results and Conclusion

Compression -Decompression
Requirement

- Compression
  - How to reverse the simplification (non parallel) ? We need
    - R (know the vertex to split)
    - P_v and P_i
    - P and C positions
    - connectivity between P , C , P_v and P_i



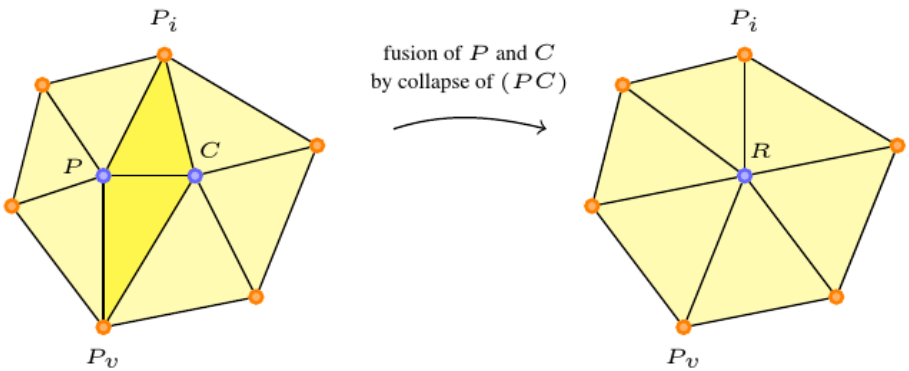




Compression -Decompression
Requirement

- How to find *R using parallelism* ?
  - Vertex valence of R decreases until it is merged with another vertex
    →*Valence of R == initial valence (creation), split is authorized*

    (one ring is at the same state as it was when R appeared, connectivity is conserved)



*To help the **compression**, all values use **the difference** with R*

Compression -Decompression
Our method : Algorithm

- Method : after entropic decompression (http)
  - *For each vertex :* find the candidate and add it to a buffer (synchronous)
  - Sort the candidates
  - *For each candidate*, if the error is too large

    P replaces R and *C* is inserted to the vertex buffer

    Two new triangles are added and connectivity in the 1 ring is updated

- Method (optimized) :
  - Sort the vertices with high priority for the candidate
  - ...

- Motivation
- State of the Art
- Parallel Simplification
- Compression – Decompression
- ***Results and Conclusion***

Results and Conclusion

- Some results

Model	Size of the connectivity information	Size per vertex
Armadillo	1.90 MB	15.2 bits
Dragon	2.07 MB	16.5 bits
France (DEM)	1.97 MB	15.8 bits

- Results are comparable to a perfect triangle strip on unsorted vertices

Results and Conclusion

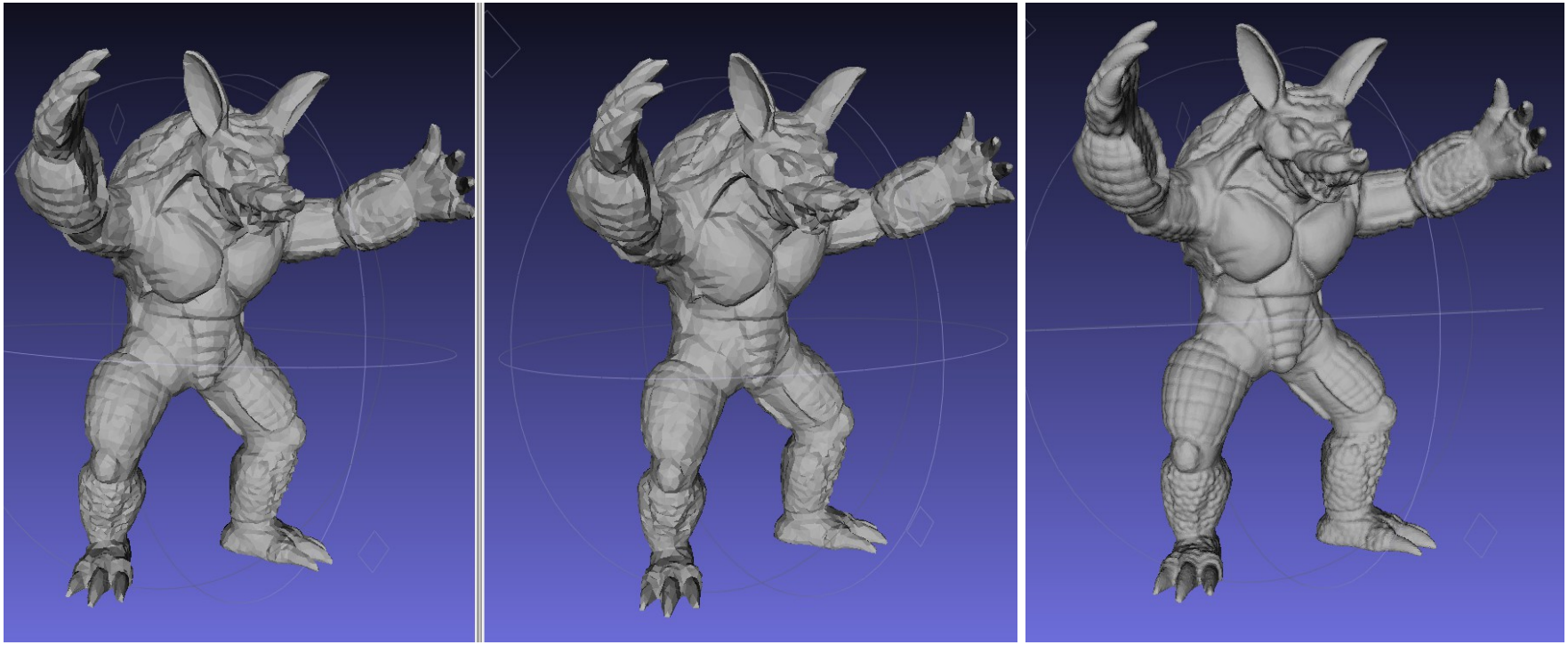
Model / format	Ply + gzip	Mply (best vertices position) + gzip	Mply (merge into one of the two vertices) + gzip
Armadillo	3.67 MB	3.91 MB	3.52 MB
Dragon	10.67 MB	11.06 MB	9.15 MB
France (DEM)	4.40 MB	4.93 MB	4.0 MB

Results and Conclusion

Garland (16 287 faces)

Our method (16727 faces)

Original (1M faces)

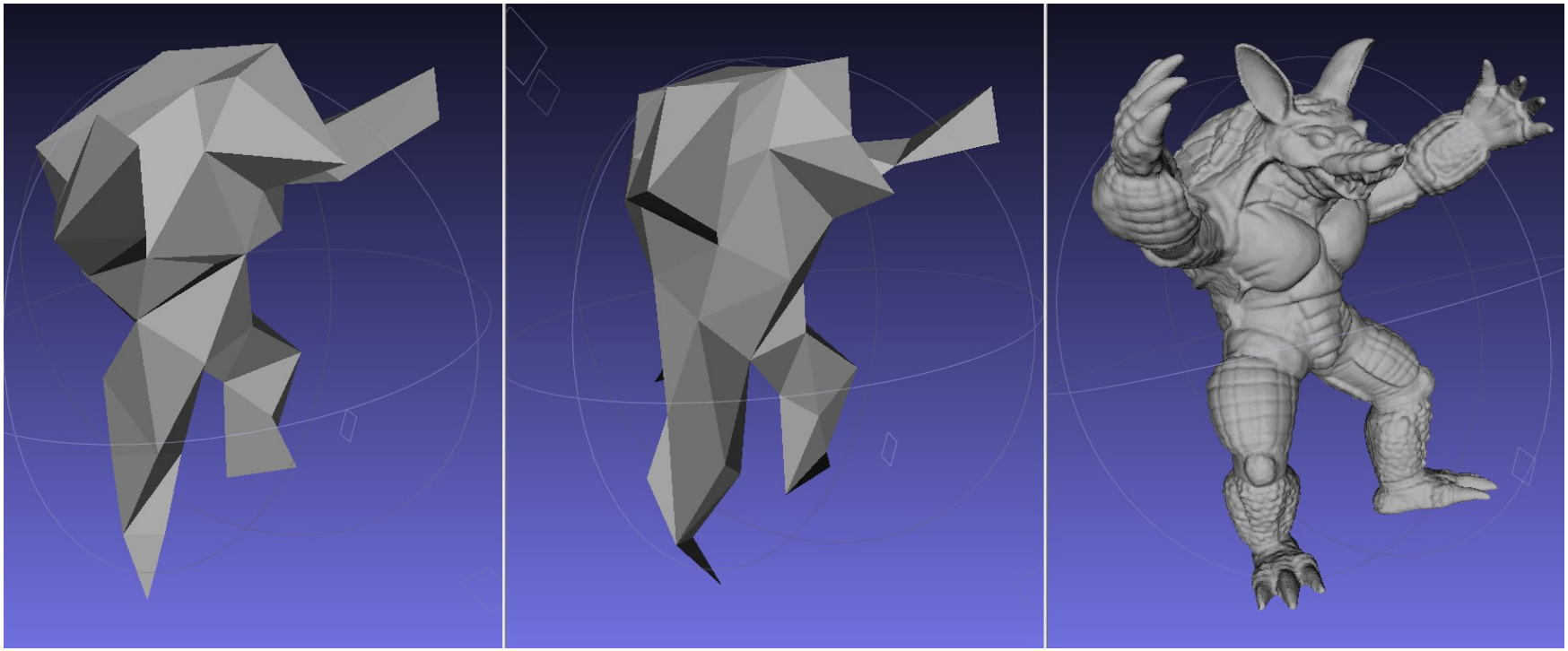


- The point of view has been chosen to show the differences

  For the same errors, our method requires approximatively 10 to 15 % more vertices for a speed up between 1.5 and 15 (intel 9200 and fx2700M)

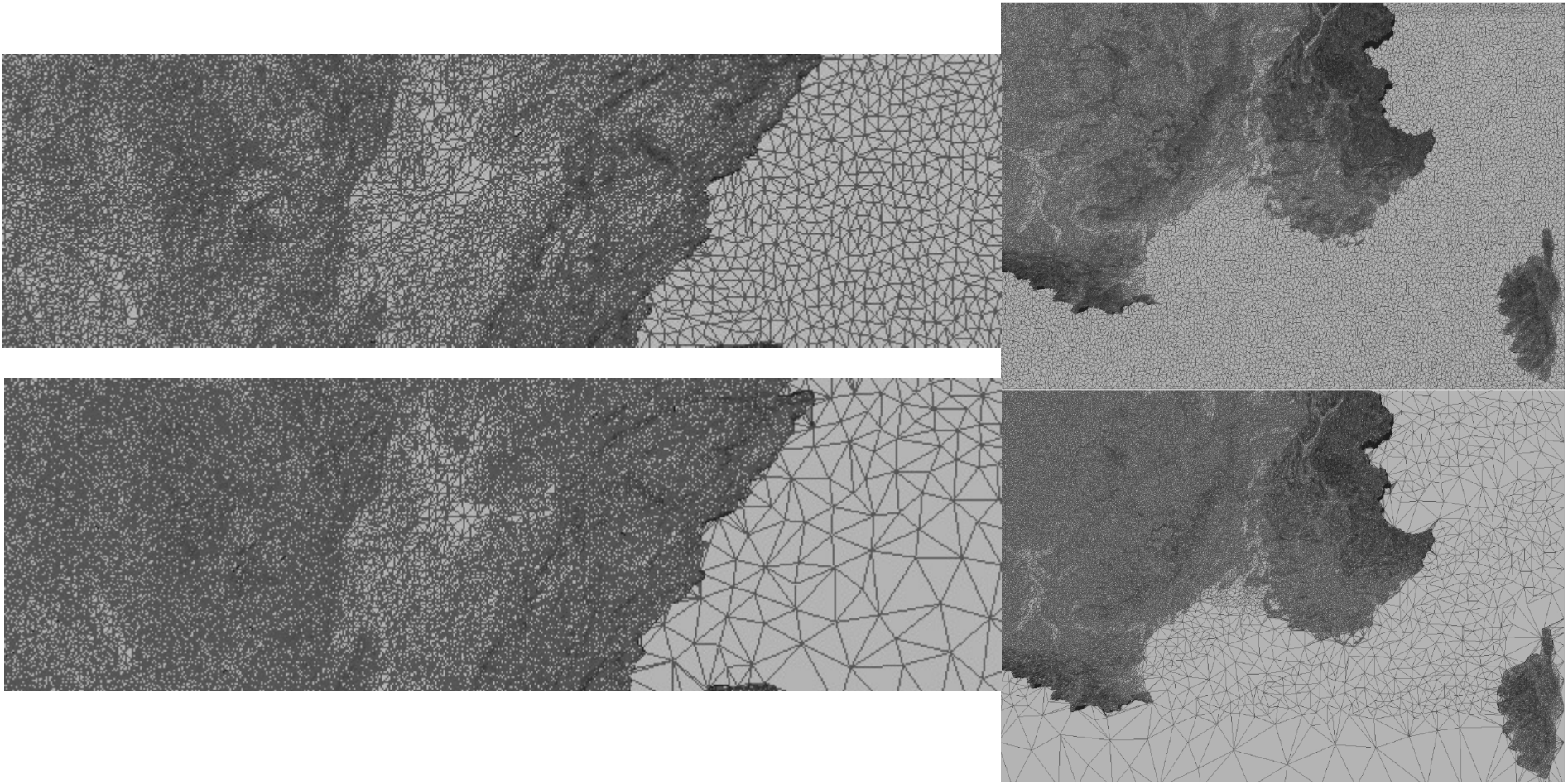
Results and Conclusion

Garland (80 faces) Our method (78 faces) Original (1M faces)



- At this stage, the number of possible choices for the fusion is too low=> error garland:4,7 * 10^{-2} and our method error 0,812 !

Results and Conclusion



- 18 iterations with high precision (threshold low) and 10 iterations with low precision to have 500k instead of 2 million vertices

Conclusion

- A parallel,unified method, streaming for all kind of models compatible with visualization (landscape, buildings, monuments, *etc.*)

- Perspectives :
  - Add textures to 3d models (easy on landscape and buildings, not for arbitrary 3d models)
  - Efficient handling of small patches
  - Parallel decompression (not only reconstruction)
  - Change the bdam segmentation with a new method which respects the model

Questions

?

?

?

?

?

?

?

?

?

?

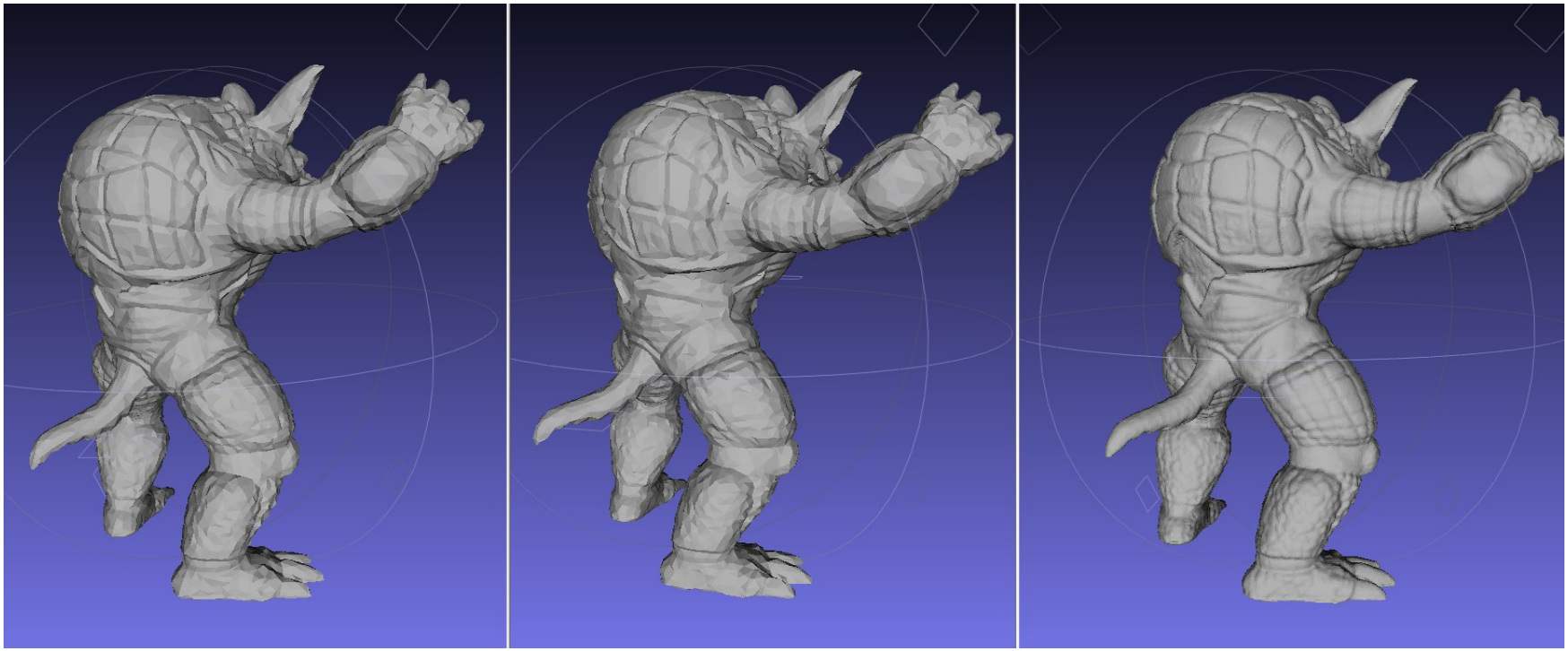
?

Stable Error
(no increase in threshold)

Garland (16 287 faces)

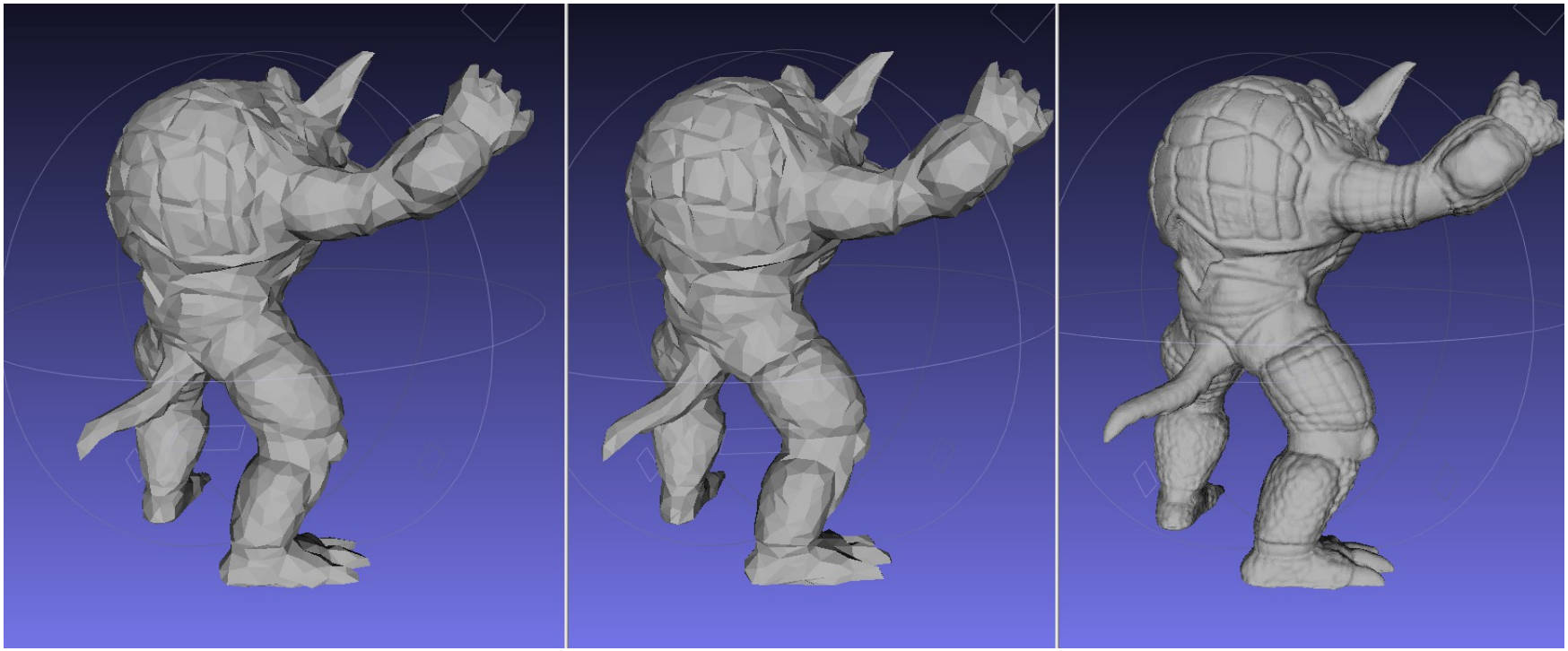
Our method (17727 faces)

Original (1M faces)



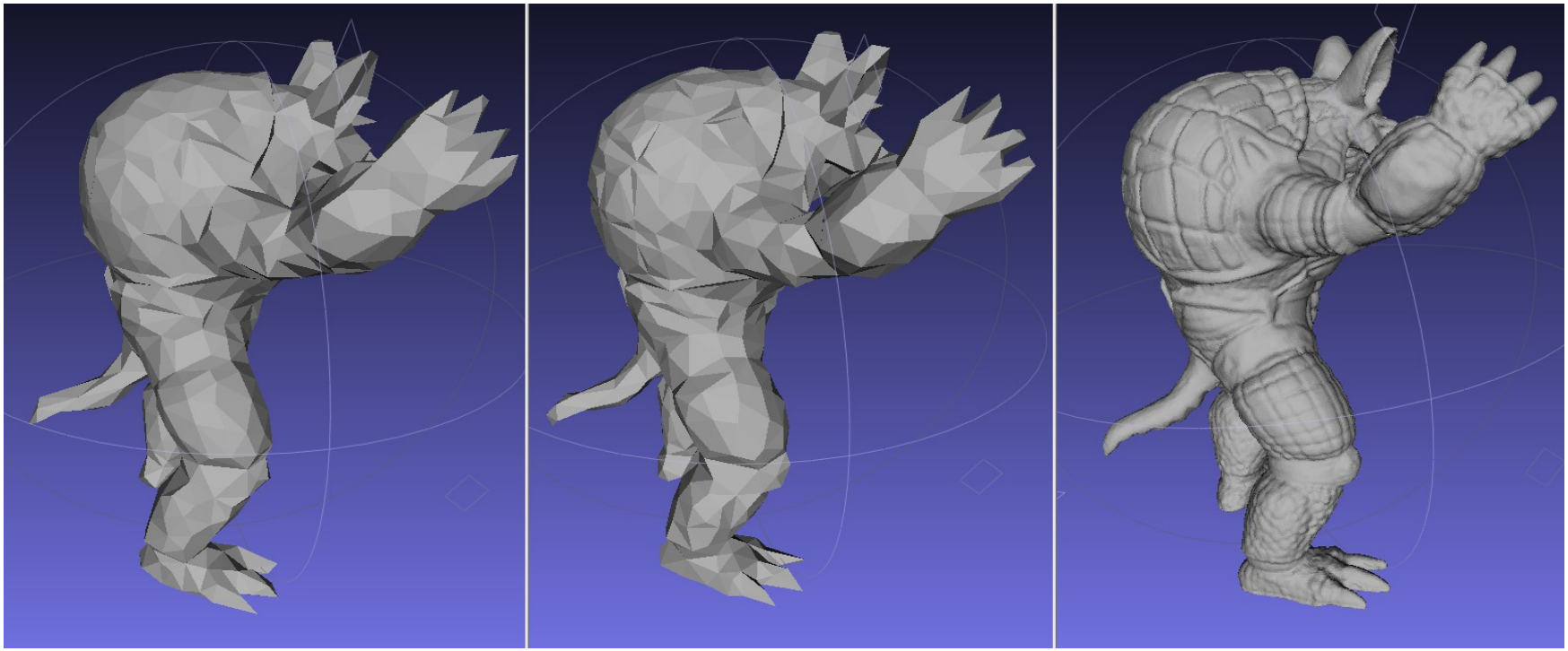
Stable Error
(no increase in threshold)

Garland (4087 faces) | Our method (4289 faces) | Original (1M faces)



Stable Error
(no increase in threshold)

Garland (2017 faces) | Our method (2420 faces) | Original (1M faces)



State of the art

View Dependent Level Of Detail

L. Hu and H. Hoppe

State of the art

View Dependent Level Of Detail
L. Hu and H. Hoppe

- Vertex fusion/split
- Hierarchy (Vt and Vu exist only if Vl and Vr exist)
- Use energy criterion for minimization :
  - Erep = energy associated to points
    (Erep decreases with the number of vertices)
  - Ep = vertex error (distance to the original vertices which generated it by fusion, with L_2 norm)
  - Edist = sum (Ep) : model error
  - Espring = sum ($||v_i - v_j||^2$)
    (adjacent vertices error to avoid thin triangles)
  - E=Espring+Edist+Erep

State of the art

View Dependent Level Of Detail
L. Hu and H. Hoppe

- Simplification creates a binary tree (the collapse of 2 vertices creates a parent node)
- Each node has an energy
- The choice of visible nodes depends on their energy and their distance from the point of view
- Not efficient for one pass decompression

State of the art

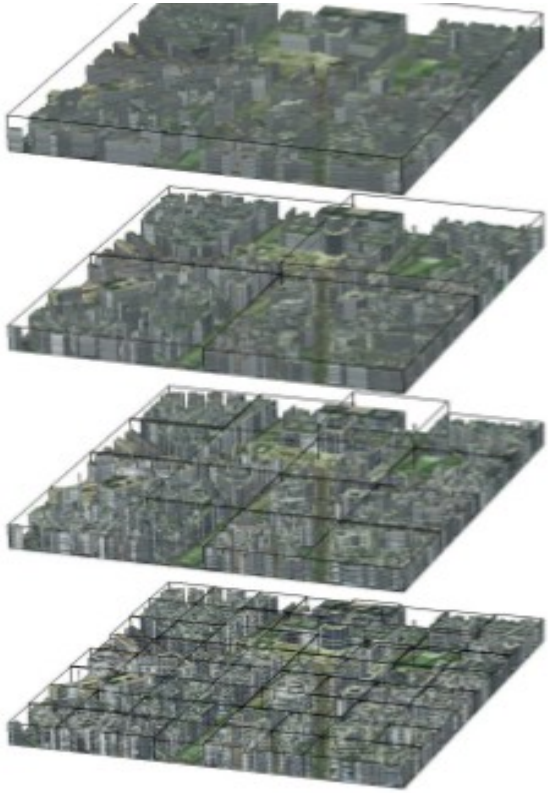
View Dependent Level Of Detail
L. Hu and H. Hoppe

- Vertices are in a common « simplification – display» memory
- 2 indices
  - One for computation
  - One for visualization
- After n steps in simplification/decompression, index swap
- The time before the next swap is predicted from the last times required to compute the n operations (to keep a good framerate)

BlockMap

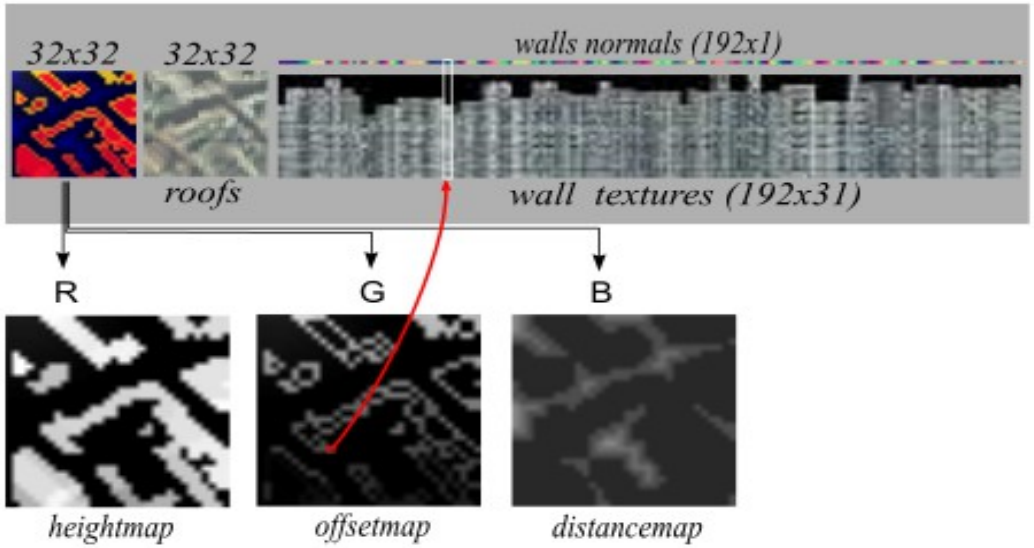
- City visualisation

- Raycasting on raster
- Each « block » allows optimisation to find the collisior

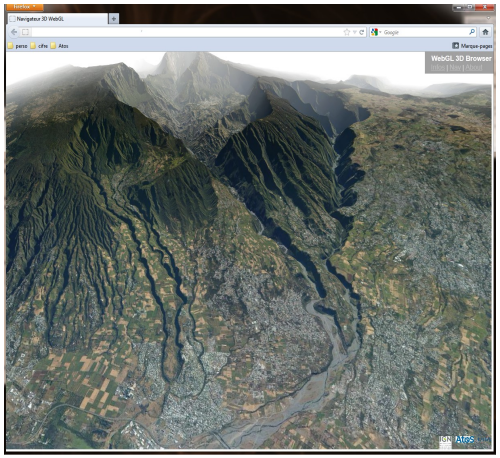


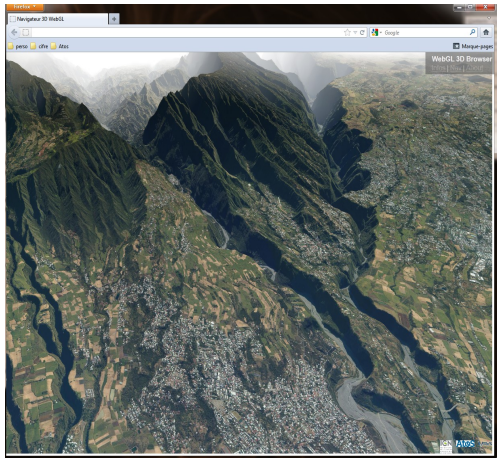
BlockMap

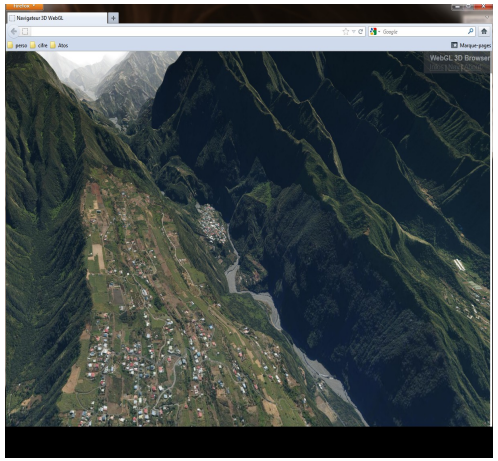
- Building are in pictures
  (ground)
- Each building knows the distance with a neighbor
  (for optimisation)
- Vertical pixel arrays are referred by an offset

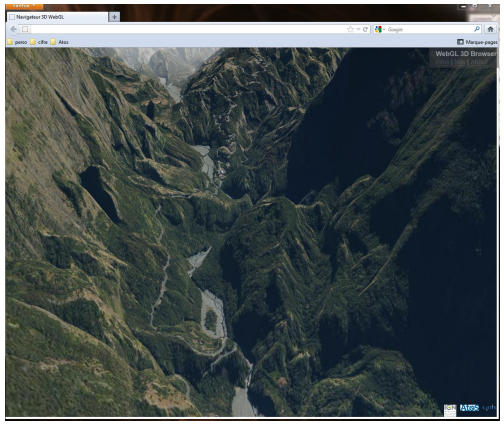


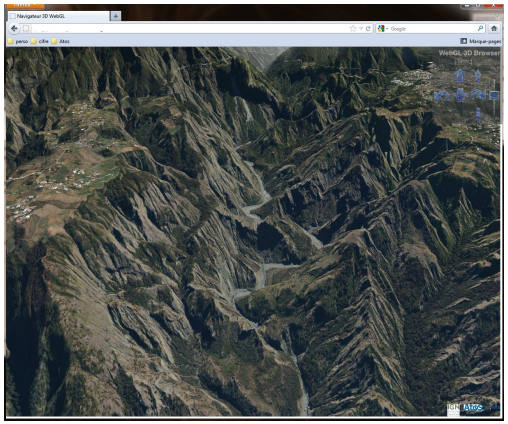
In OpenscalesGL

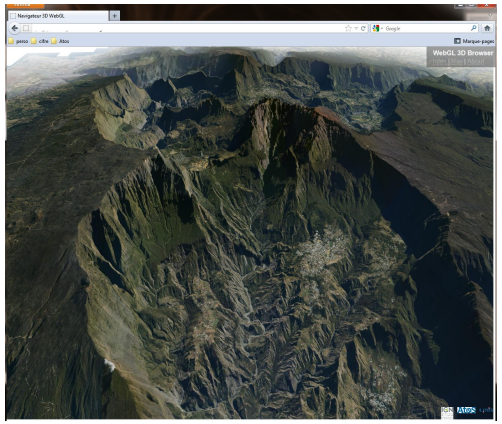


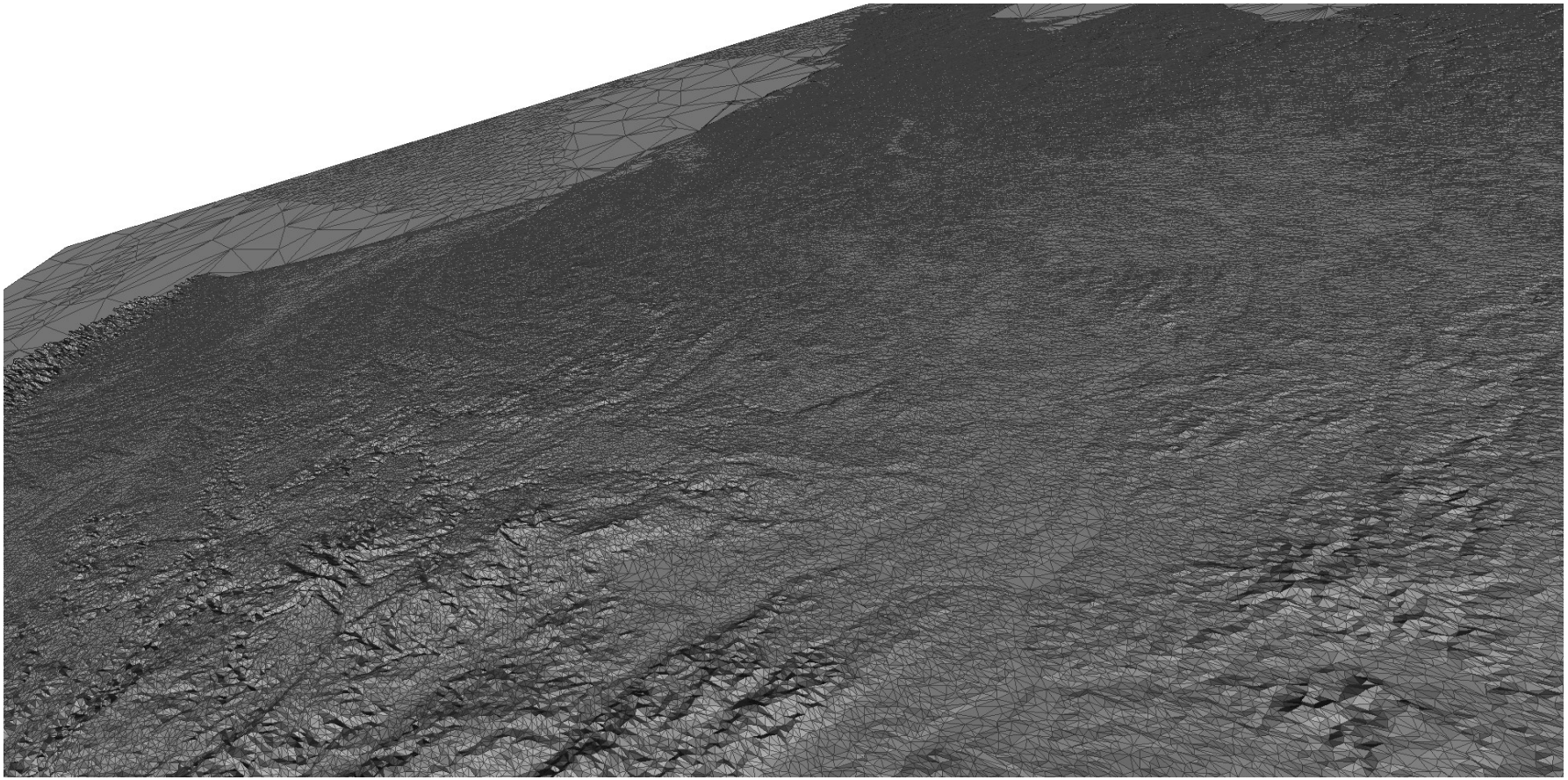


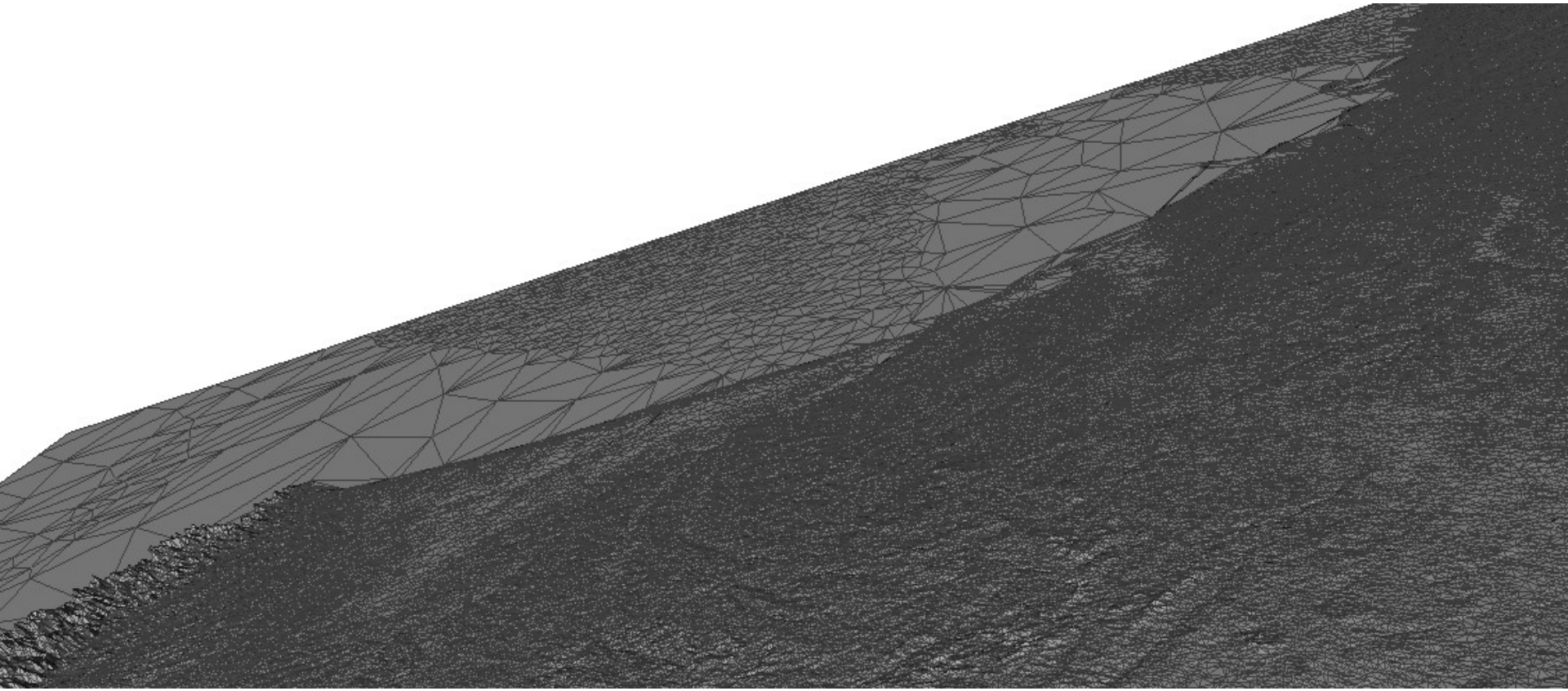


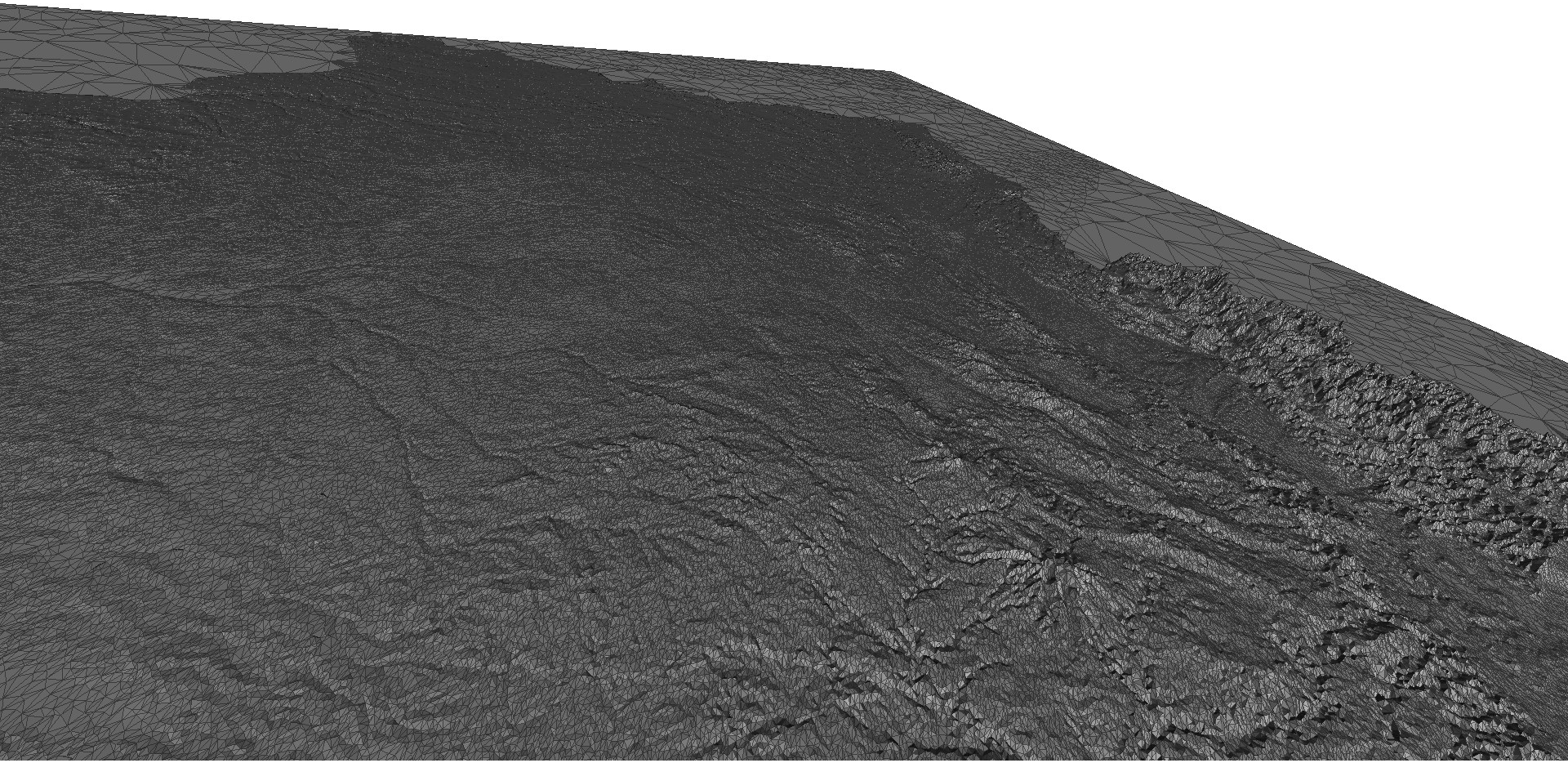












Global compression

Model / format	Ply + gzip	Mply (best vertices position) + gzip	Mply (merge into one of the two vertices) + gzip
Armadillo	3.67 MB	3.91 MB	3.52 MB
Dragon	10.67 MB	11.06 MB	9.15 MB
France (DEM)	4.40 MB	4.93 MB	4.0 MB