# ViZiR 4: visualization of high-order meshes and solutions software

**Matthieu Maunoury, Adrien Loseille**
Inria Saclay, GAMMA team, France
Matthieu.Maunoury@inria.fr, Adrien.Loseille@inria.fr
International Meshing Roundtable 2023, Amsterdam, Netherlands

## Introduction

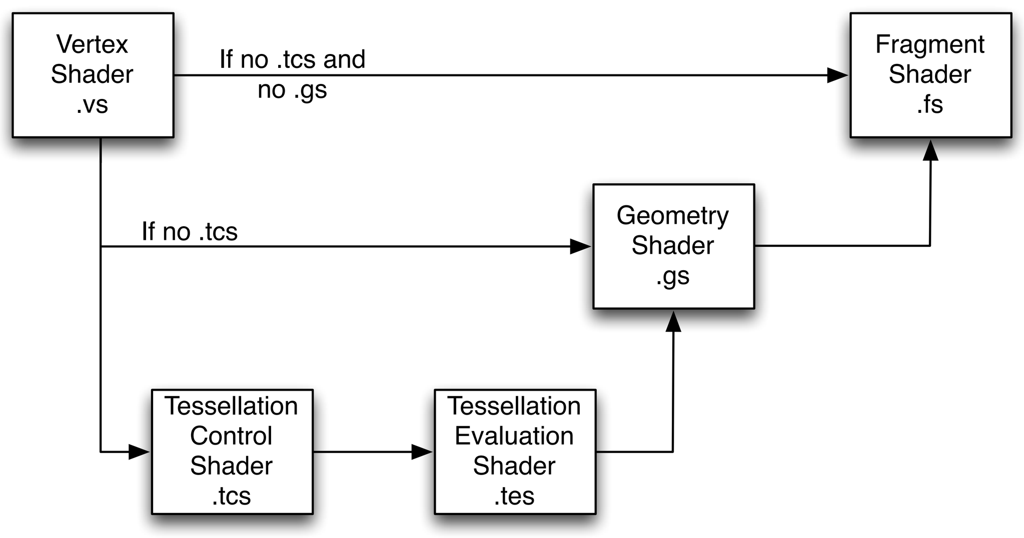
Main features of `ViZiR 4`:

- Light, simple and interactive visualization software.
- Surface and volume (tetrahedra, pyramids, prisms, hexahedra) meshes.
- Pixel exact rendering of high-order solutions on straight elements.
- Almost pixel exact rendering on curved elements (high-order meshes).
- Post-processing tools, such as picking, isolines, clipping, capping.

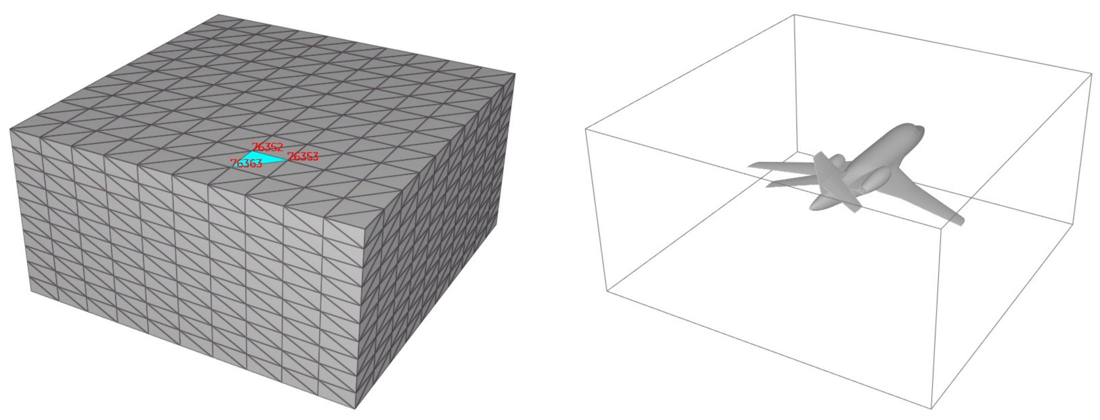
Visit `http://vizir.inria.fr` to download and try `ViZiR 4`!

## OpenGL 4 graphic pipeline

OpenGL Shading Language (GLSL) allows to perform pixel exact rendering of high order solutions on straight elements and almost pixel exact rendering on curved elements. The description of the shaders (see [1, 2] for more details) is the following:




- **Vertex Shader** (VS). It transforms vertex positions into clip space.
- **Fragment Shader** (FS). It determines the appropriate color for each pixel. It leads to pixel exact rendering. FS is used to perform high-order solutions and isolines rendering.
- **Geometry Shader** (GS). It modifies the thickness of lines for edges, it also computes the distance of tessellated vertices to the clip plane.
- **Tessellation Control Shader** (TCS). Elements are subdivided (i.e. tessellated). This level of discretization is controlled by geometrical error estimated and can be refined or unrefined on the fly.
- **Tessellation Evaluation Shader** (TES). For each vertex of the subdivision, it determines its position in the physical space.



(a) Picking (selection) an element (b) Hiding elements sharing same reference

Figure 1: Any element or vertex can be picked to have info (index, coordinates, indices of vertices, solutions values...) and surfaces sharing the same reference id can be hidden.

A key to have an efficient visualization is to quickly open meshes and solutions files. Input and output are handle by the `libMeshb` library. For instance, the mesh of the windmill (see Figure 2) with 3.3 millions vertices and 6.4 millions triangles is opened in 0.05 seconds and the total time to render it is 0.7 seconds.

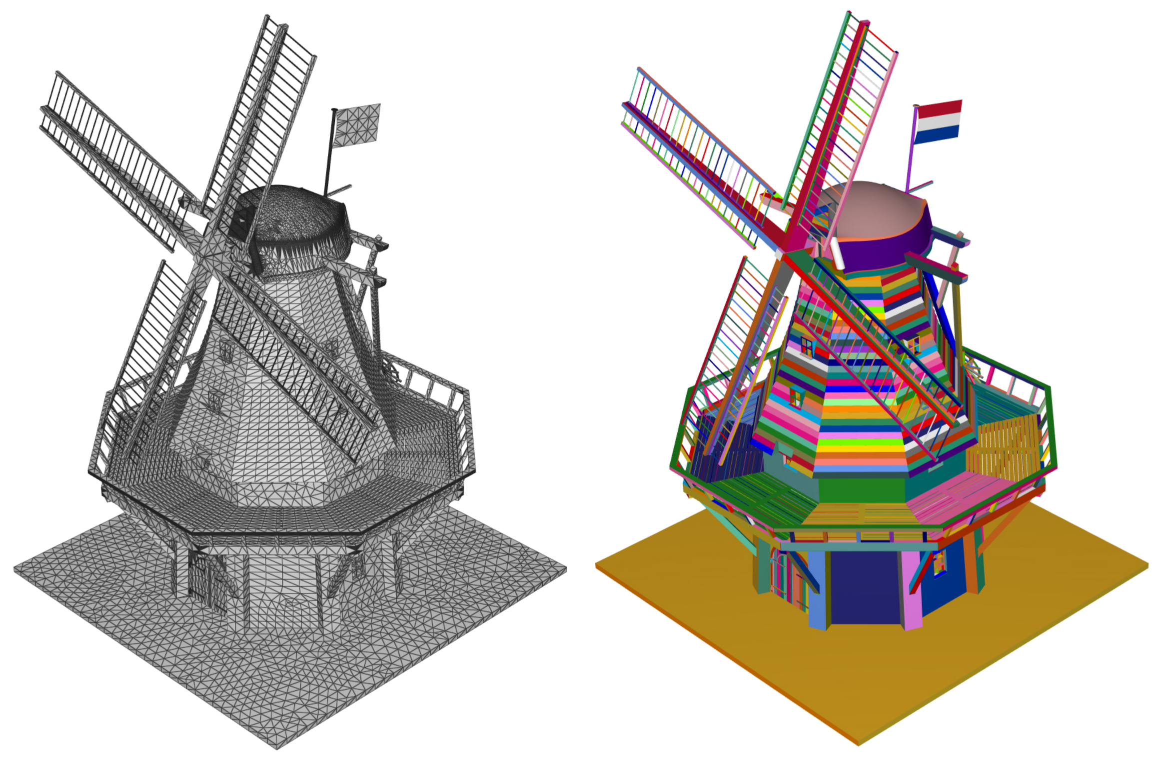


Figure 2: A windmill mesh (left) and references (right).

## Pixel exact rendering on straight elements

OpenGL 4 graphic pipeline flexibility leads to a faithful and interactive pixel exact rendering (without visualization error) when straight elements (of degree one) are considered regardless of the degree of the solution (see Figure 3).

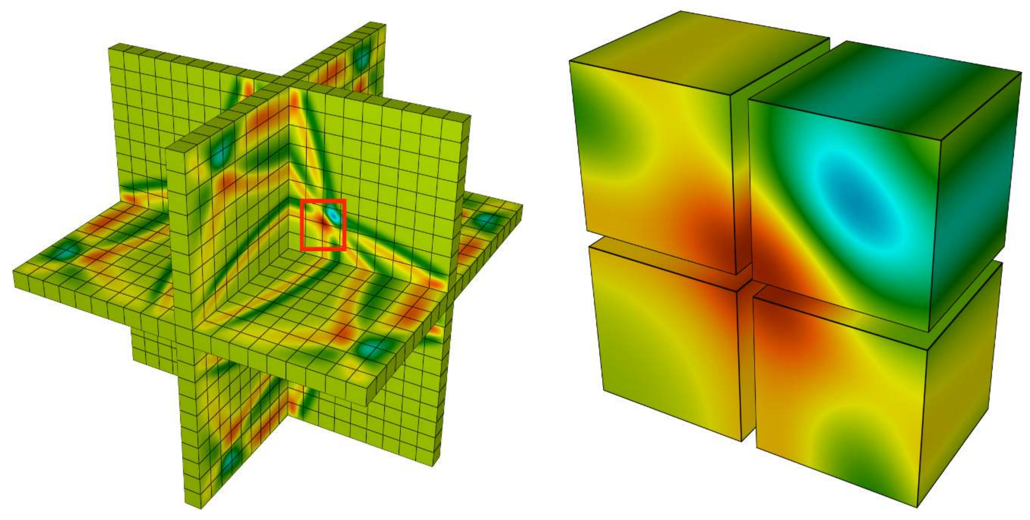
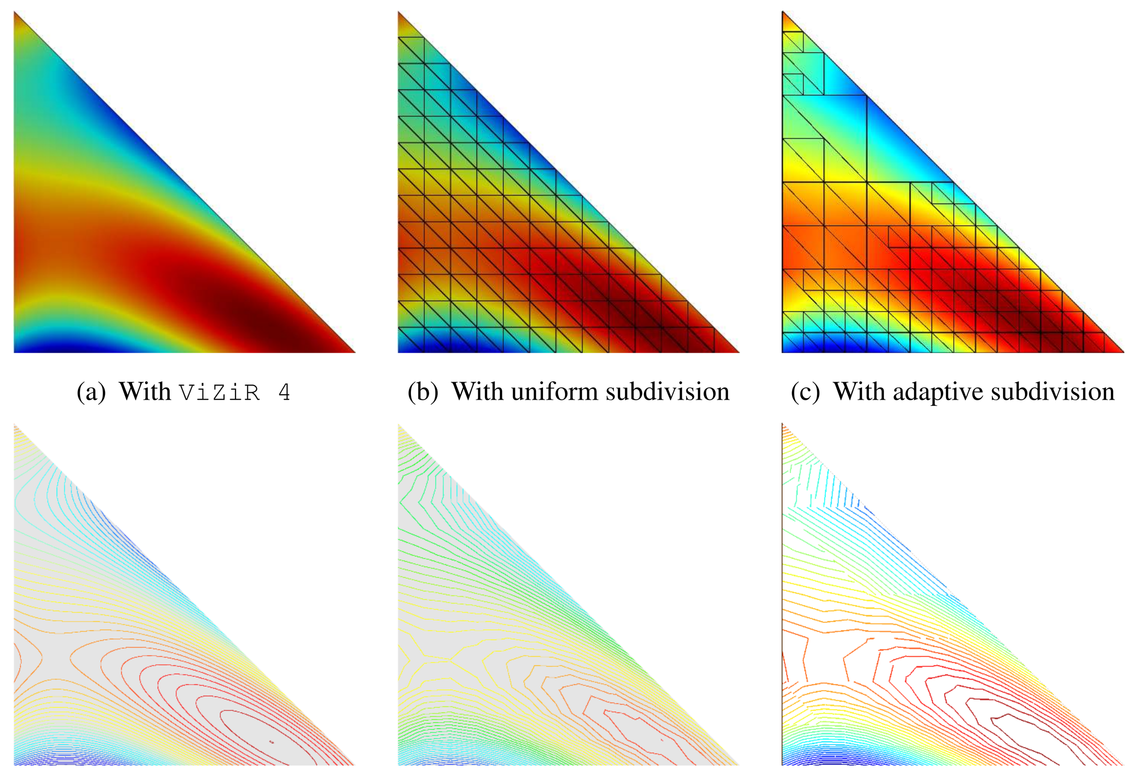


Figure 3: Illustration of pixel exact rendering for a $Q^6$ solution. Zoom in 4 hexes (right).

Figure 4 compares pixel exact rendering of `ViZiR 4` to "classic" affine representations. A visualization error appears when subdivisions are used to generate a $P^1$ representation by "classic" approaches.

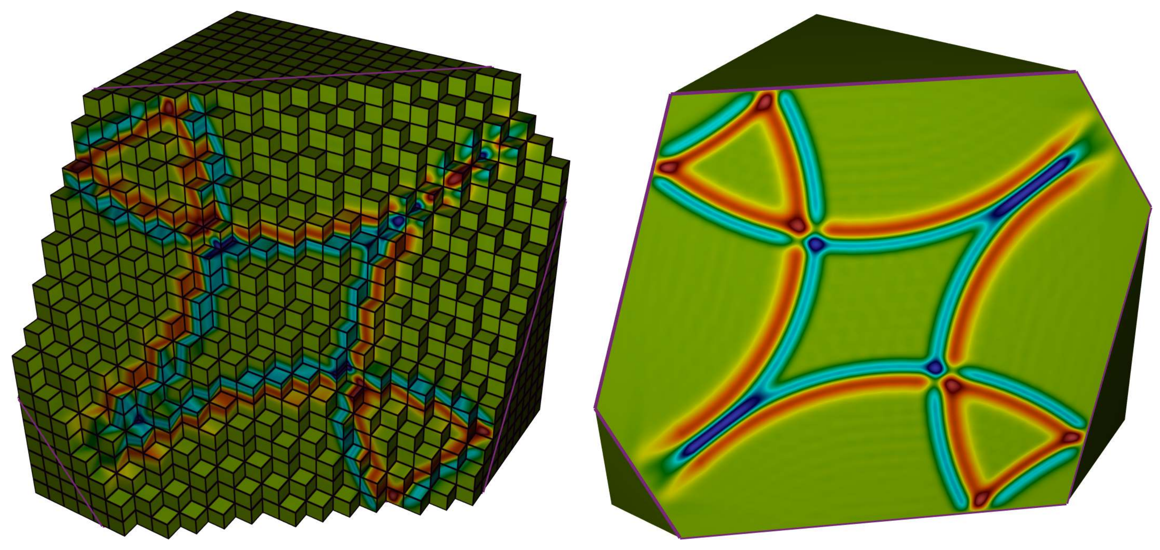


(a) With `ViZiR 4` (b) With uniform subdivision (c) With adaptive subdivision

(d) With `ViZiR 4` (e) With uniform subdivision (f) With adaptive subdivision

Figure 4: Rendering (top) and isolines (bottom) of a $P^3$-polynomial on a single triangle.
Left: pixel exact rendering with `ViZiR 4`.
Middle: uniform subdivision of 169 triangles.
Right: adaptive subdivision of 169 triangles.

All post-processing tools such as clipping or capping, are interactive in `ViZiR 4`: the plane can be rotated or translated on the fly (Figure 5).



(a) Clipping (the mesh is displayed) (b) Capping (intersecting the cut plane)

Figure 5: Clipping (a) and capping (b) for a $Q^6$ solution on a mesh of 8000 hexahedra.

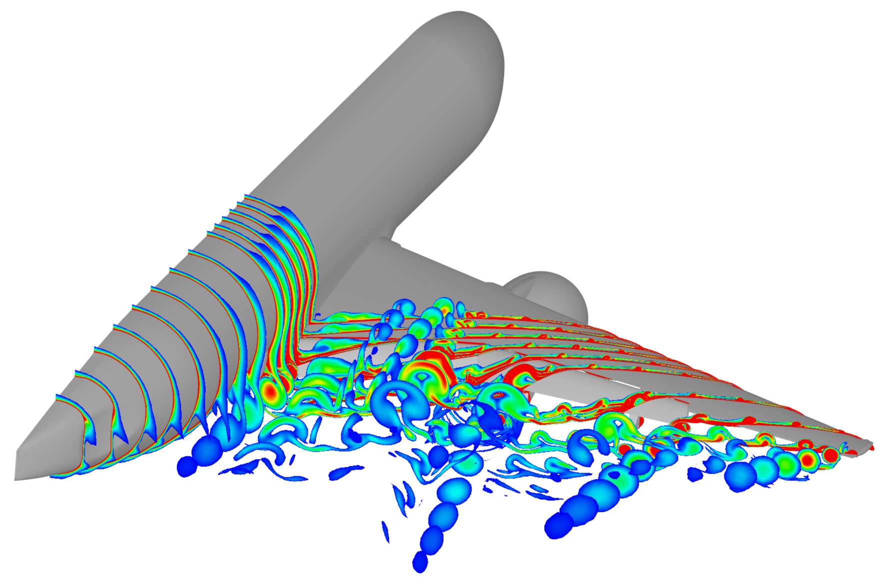


Figure 6: Several user planes have been defined to display the solution in the volume. A cutoff is done to discard pixels whose values are outside a given range.

## High order elements

For more complex geometries, curved elements perform a better approximation of the geometry (see Figure 7).

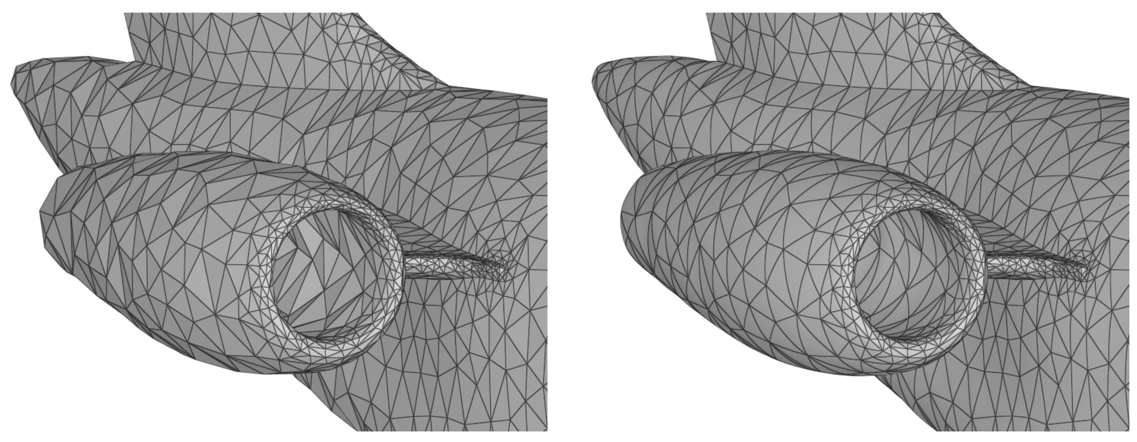


Figure 7: Comparison of a $P^1$ mesh (left) and a $P^3$ mesh (right).

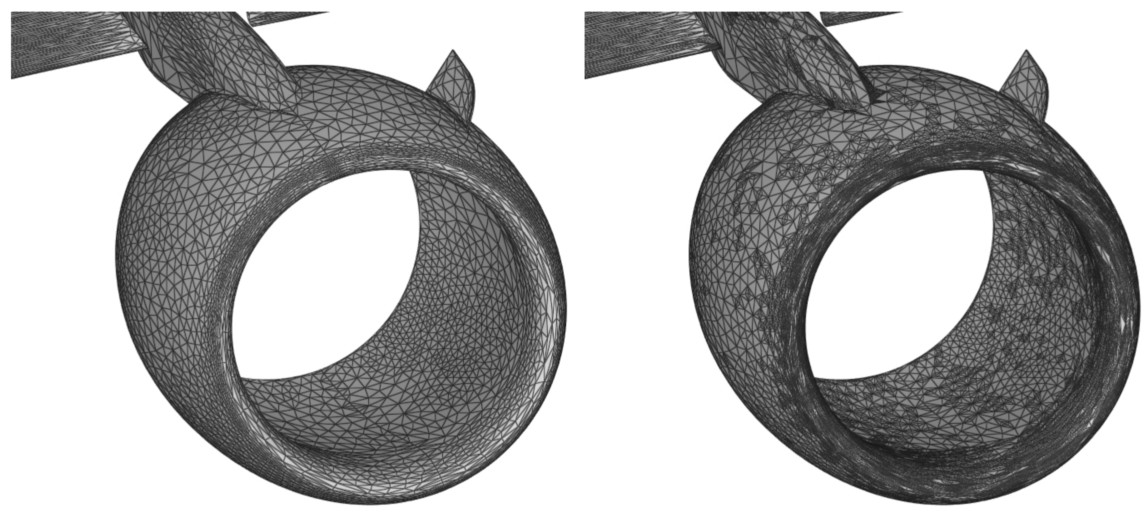


Figure 8: Rendering of a $P^3$ mesh (left) and its tessellation (right) done by the shaders (directly on the GPU).

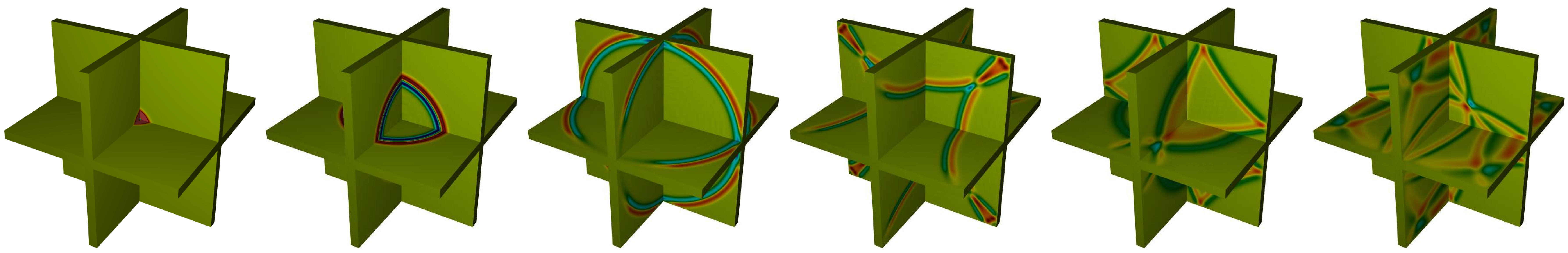


Figure 9: $Q^6$ solution of a wave propagation problem at different times.

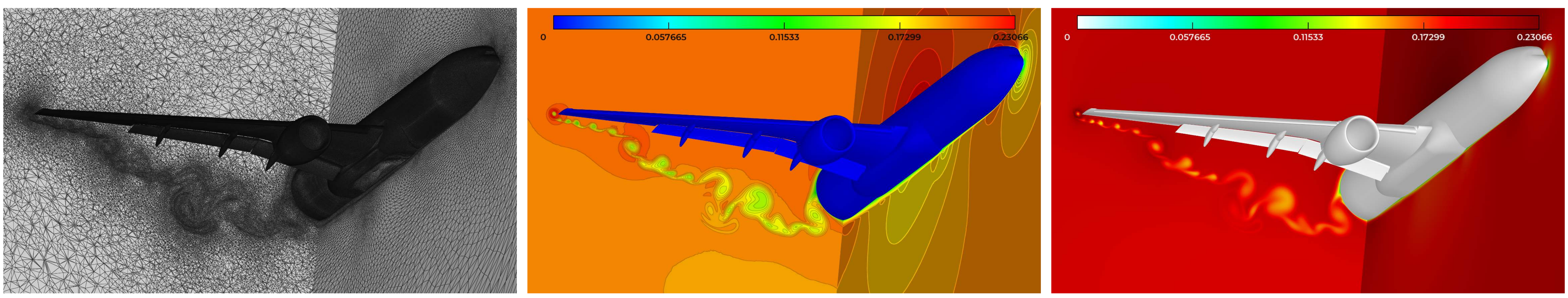



Figure 10: Example of adaptive high-lift flows around the 4th AIAA CFD High-Lift Prediction Workshop geometry (CRM-HL geometry). The adapted mesh is composed of 20 millions vertices and 120 millions tetrahedra and the local Mach number is the solution field.

## References

[1] R. Feuillet, M. Maunoury, and A. Loseille. On pixel-exact rendering for high-order mesh and solution. *Journal of Computational Physics*, 424:109860, 2021.
[2] A. Loseille and R. Feuillet. Vizir: High-order mesh and solution visualization using opengl 4.0 graphic pipeline. *56th AIAA Aerospace Sciences Meeting, AIAA Scitech*, 2018.