Visualization of large meshes and solutions from numerical simulations with ViZiR 4

Matthieu Maunoury, Adrien Loseille

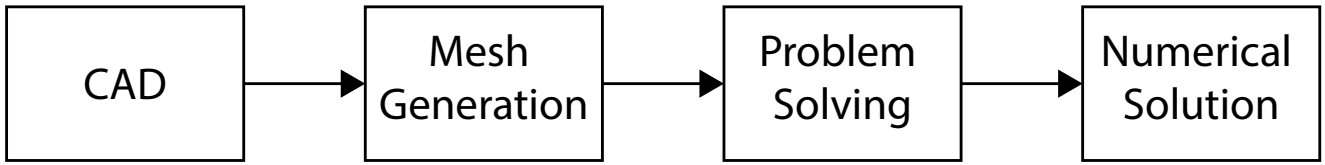
Inria Saclay
GAMMA (Automatic Mesh Generation and Adaptation Methods) Team

Journée Visu 2021
June 08, 2021

Introduction

Numerical simulations: predict the behavior of physical phenomenons without using prototypes or experimentations.

Applications: Computational Fluid Dynamic, acoustics, energy, electromagnetism or medical modeling...



Visualization tools are necessary:

- inspect the CAD model
- check the validity and quality of the meshes
- display the numerical solutions computed
- analyze the potential problems on meshes and solutions
- validate algorithms

Visualization softwares

Many visualization softwares (e.g. Gmsh, Medit, ParaView, Tecplot, VisIt, Vizir Legacy) to analyze numerical results:

- Based on **linear** primitives as imposed by the commonly-used baseline graphic pipeline.
- Many interesting **plugins** and **tools** to help the analyses.

Some limitations:

- **Interactivity** might be missing (time to open files and render meshes and solutions).
- Lack of tools to **manipulate efficiently** these meshes.
- High-order meshes are **generally not handled**.
- High-order solutions: **visualization error** due to *low order remeshing*.

Interactivity bottleneck: CPU Times comparisons

# vertices	# triangles	# tetrahedra	ParaView (s)	ViZiR 4 (s)	Ratio
1 342 310	446 158	7 370 829	14.9	0.76	**19.6**
2 699 131	802 316	14 968 807	32.8	1.20	**27.3**
5 415 482	1 285 472	30 541 700	71.4	1.80	**39.7**
10 784 310	2 080 672	61 563 158	155.1	3.12	**49.7**
21 695 268	3 614 018	124 736 423	333.6	7.78	**42.9**
43 380 172	6 275 672	250 898 971	980.2	16.05	**61.1**
3 084 324	6 166 689	0	11.2	2.25	**5.0**

Table 1: Comparison of total rendering CPU time (s) including files (mesh and solution) opening.

# vertices	# triangles	# tetrahedra	ParaView (s)	ViZiR 4 (s)	Ratio
1 342 310	446 158	7 370 829	1.6	0.18	**8.9**
2 699 131	802 316	14 968 807	3.1	0.48	**6.5**
5 415 482	1 285 472	30 541 700	6.4	0.93	**6.9**
10 784 310	2 080 672	61 563 158	13.9	1.99	**7.0**
21 695 268	3 614 018	124 736 423	28.6	4.80	**6.0**
43 380 172	6 275 672	250 898 971	91.3	10.04	**9.1**

Table 2: Comparison of CPU time (s) to generate cut plane (clip).

Interactivity bottleneck: CPU Times comparisons

Case	# vertices	# triangles	ParaView (s)	ViZiR 4 (s)
640K	217 000	433 653	7.2	0.01
1280K	392 257	783 947	13.9	0.01
2560K	628 223	1 255 604	24.6	0.01
5120K	1 018 135	2 035 092	39.2	0.01
10240K	1 772 712	3 543 955	63.1	0.01
20480K	3 084 324	6 166 689	109.3	0.01

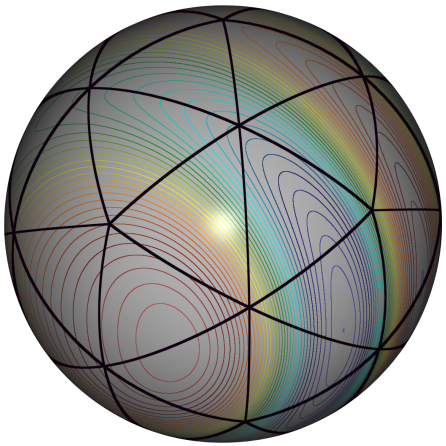
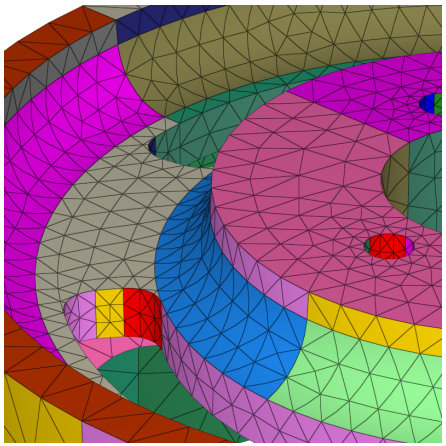
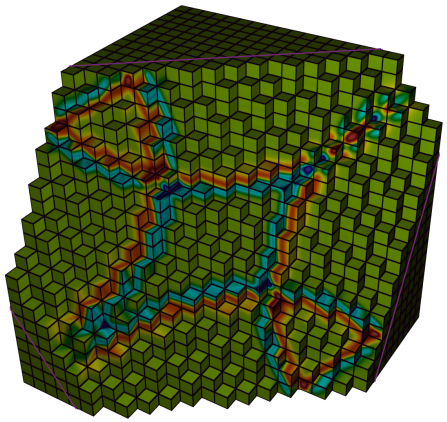
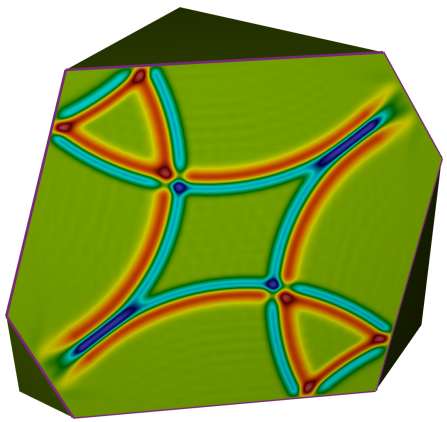
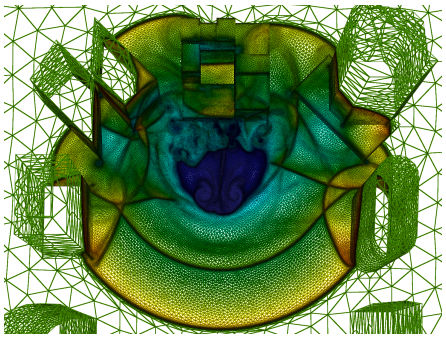
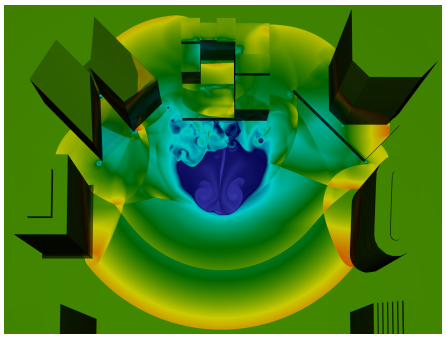
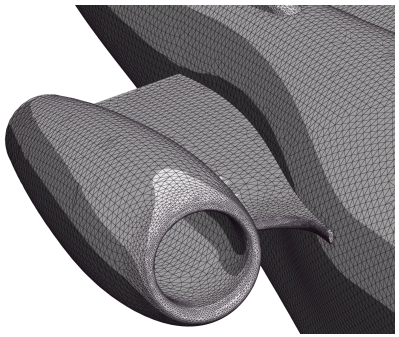
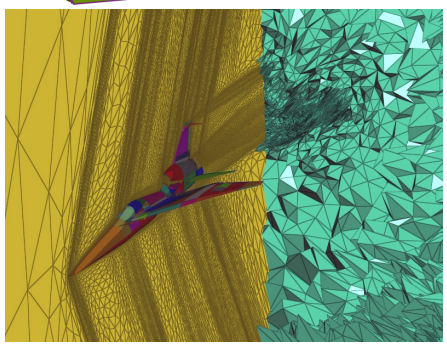
Table 3: Comparison of CPU time (s) to render isolines (contours) for different meshes composed only of triangles.

Some issues ViZiR 4 tries to answer:

- **Interactivity** (i.e. fast) to develop meshes algorithms.
- Display with **high fidelity** the computed numerical solution.
- Handle **high-order** meshes and solutions.

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.

OpenGL 4 graphic pipeline

`ViZiR 4` is based on OpenGL Shading Language (GLSL).

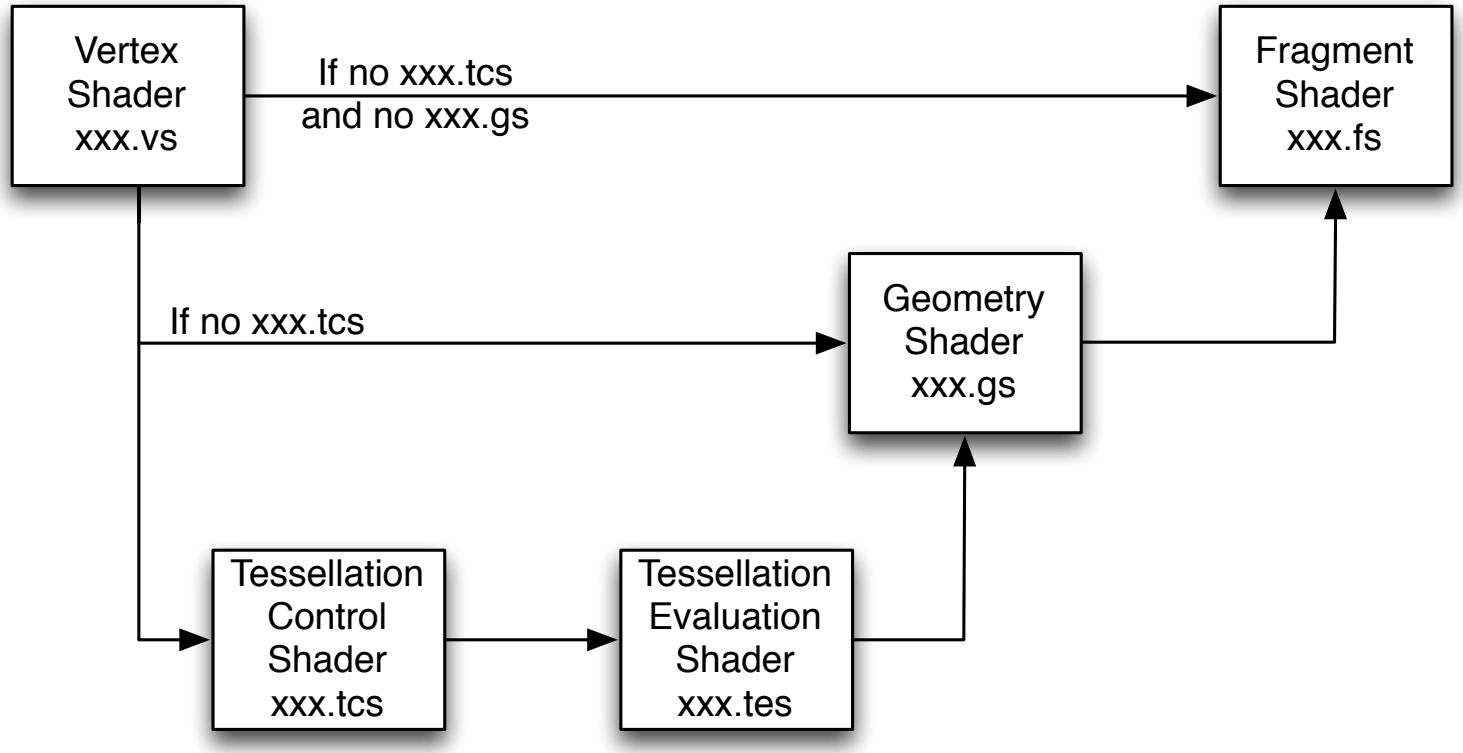
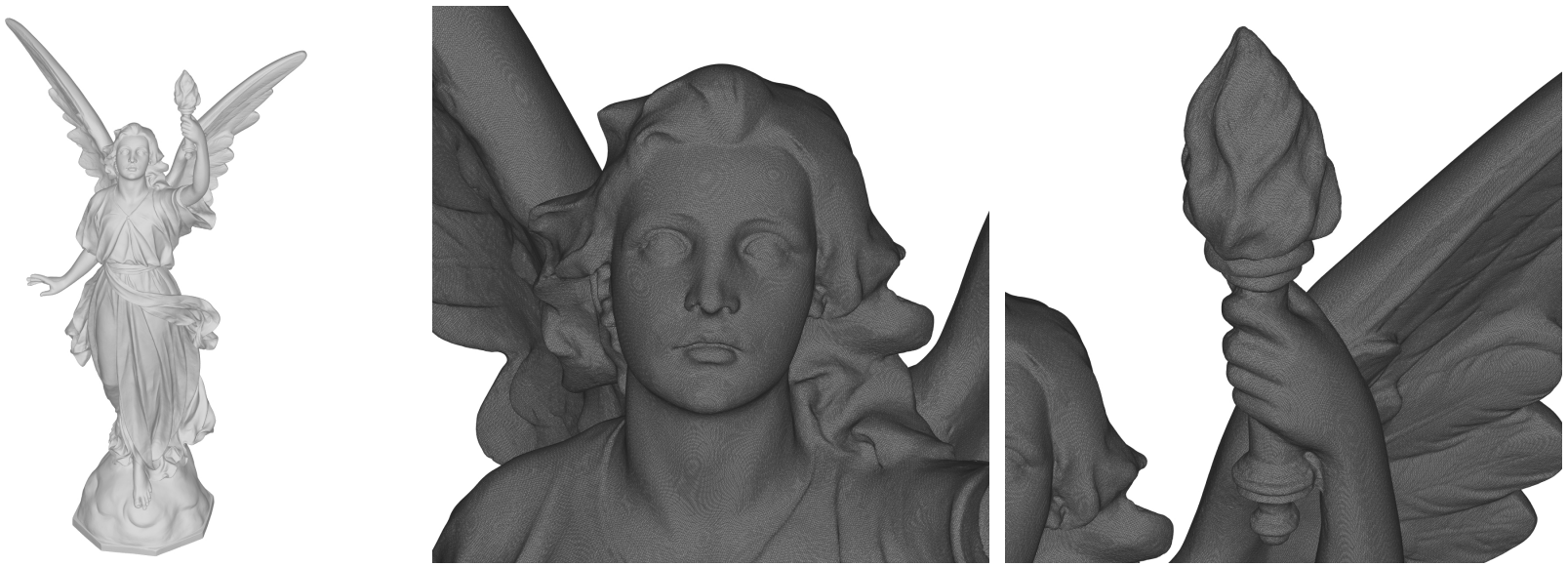


Figure 1: Shaders used for the OpenGL graphic pipeline.

Fast I/O

Input and output handle by the libMeshb library (Loic Maréchal, Inria).



Mesh of Lucy:

- 14 millions vertices and 28 millions triangles (642 Mb).
- Mesh opened in less than 1.5 seconds.
- Rendered in 7.5 seconds (total time) on a laptop.

Pixel exact rendering on flat elements

- For each pixel, Fragment shader determines the appropriate color.
- It certifies a faithful and interactive depiction (up to degree 10 polynomial function).
- High order solutions are natively handled by `ViZiR 4` on surface and volume (tetrahedra, pyramids, prisms, hexahedra) meshes.

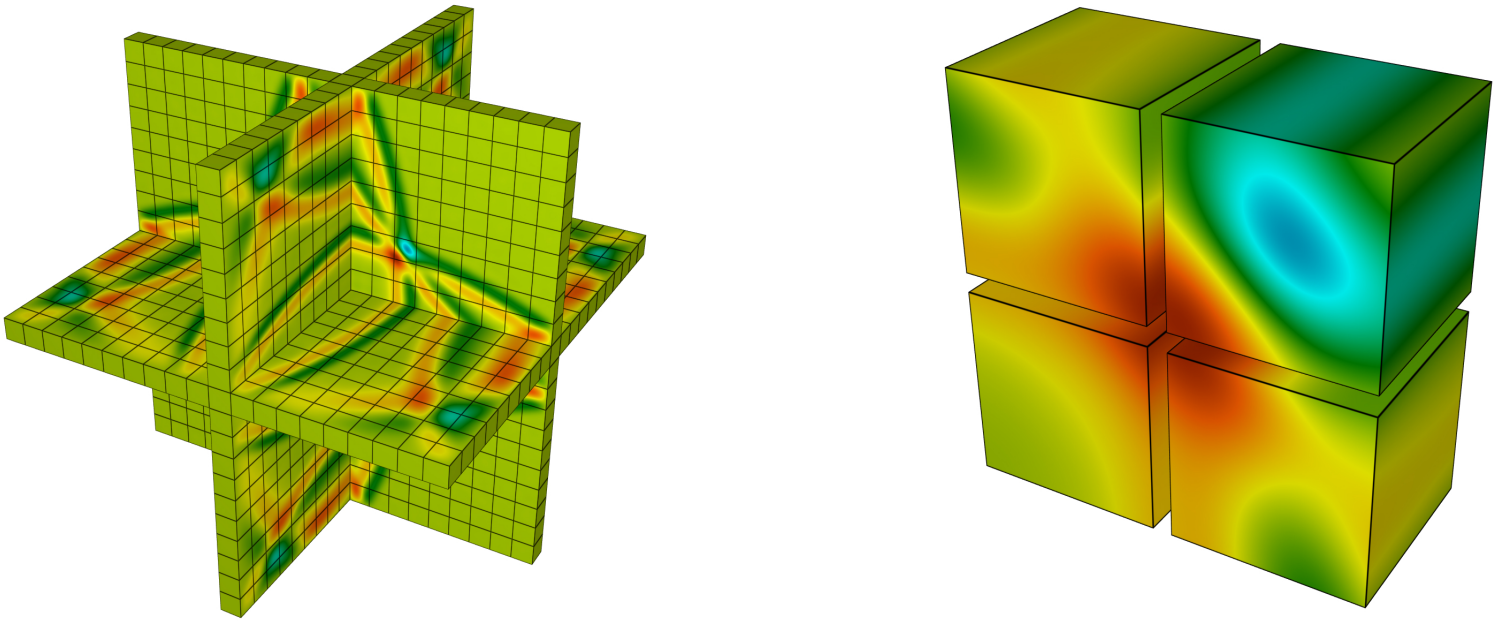


Figure 2: High-order (degree Q^6) solution of a wave propagation problem. Right: zoom of the solution on 4 hexahedra.

Tessellation on GPU for high-order elements

- For complex geometry, curved elements perform a better approximation of the geometry.
- Tessellation shaders: creation of sub-elements **on the fly** by the GPU.

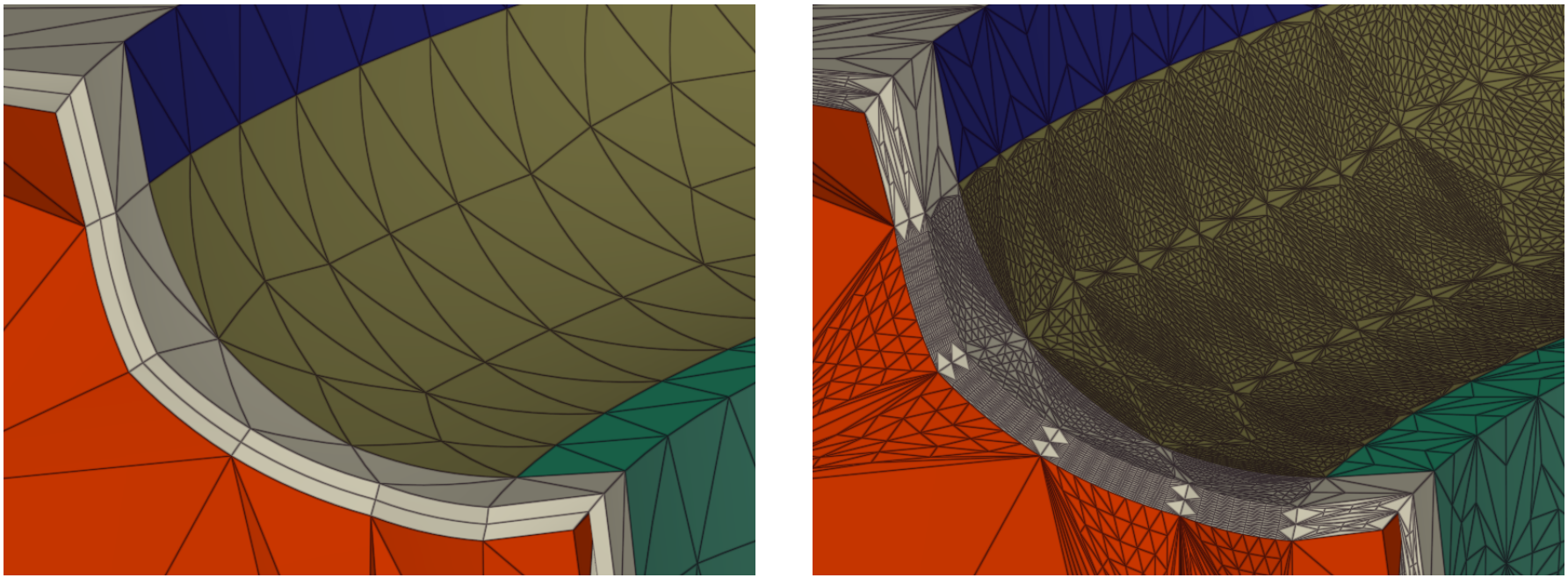


Figure 3: Rendering of high-order mesh (left) and its tessellation constructed by the GPU (right).

Tessellation on GPU for high-order elements

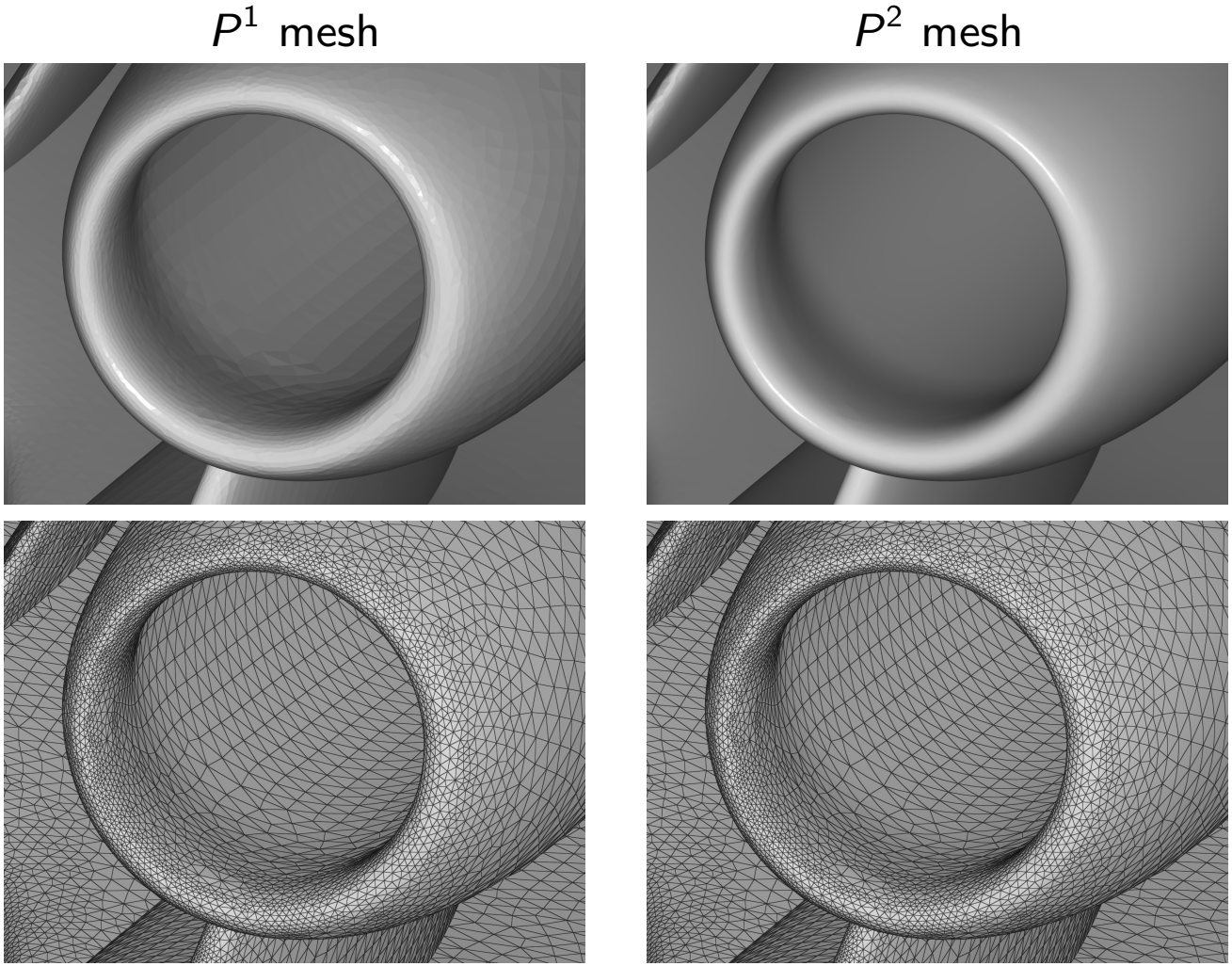


Figure 4: Comparison of meshes of degree 1 (left) and 2 (right) for the same number of elements.

Isolines (**instant** rendering) with possibly filled solution rendering

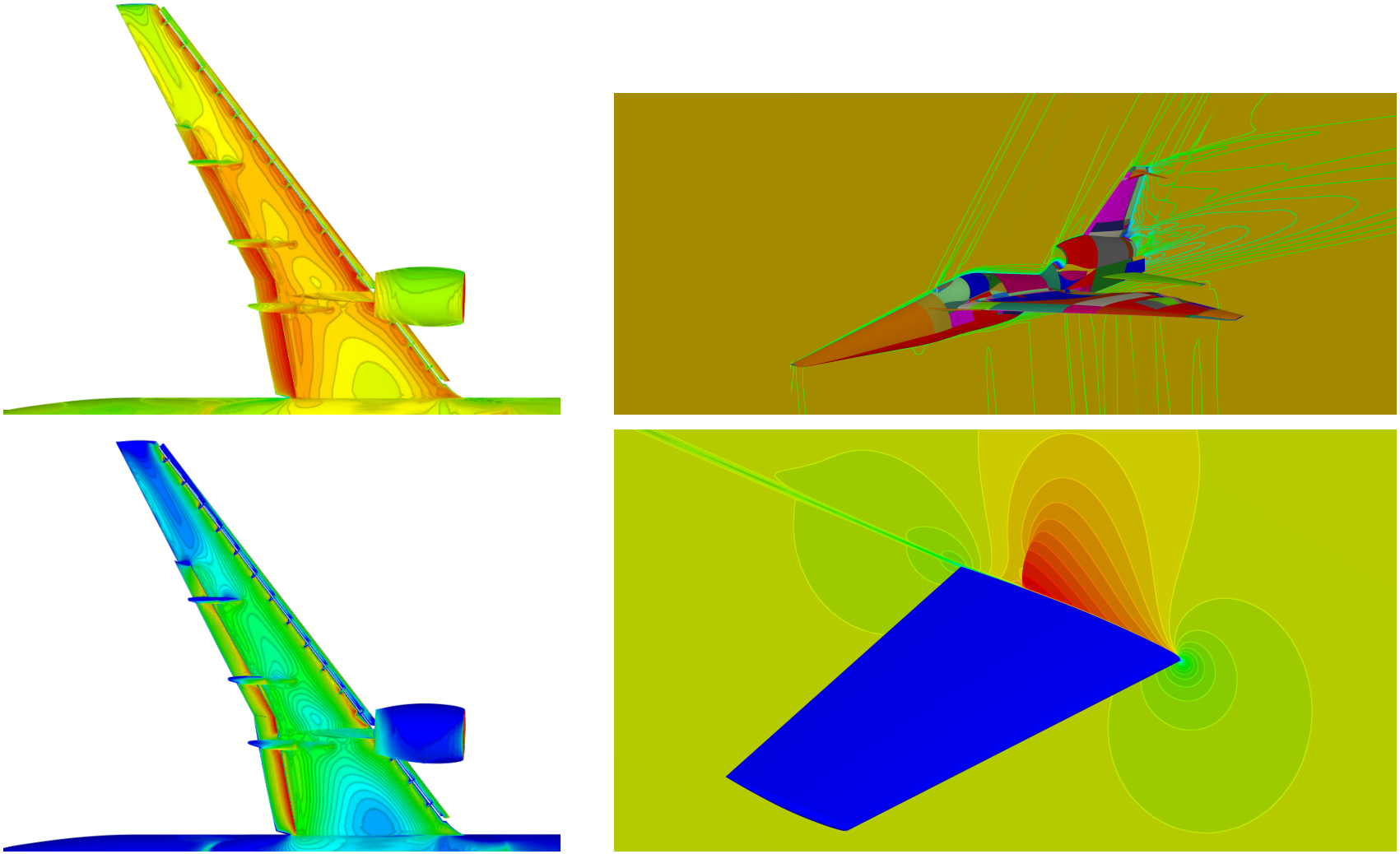


Figure 5: Examples of isolines rendering.

Clip Planes

- Difficult to navigate in 3 dimensional meshes.
- For this reason, interesting to use clip planes where all volume elements belonging to a plane are displayed.

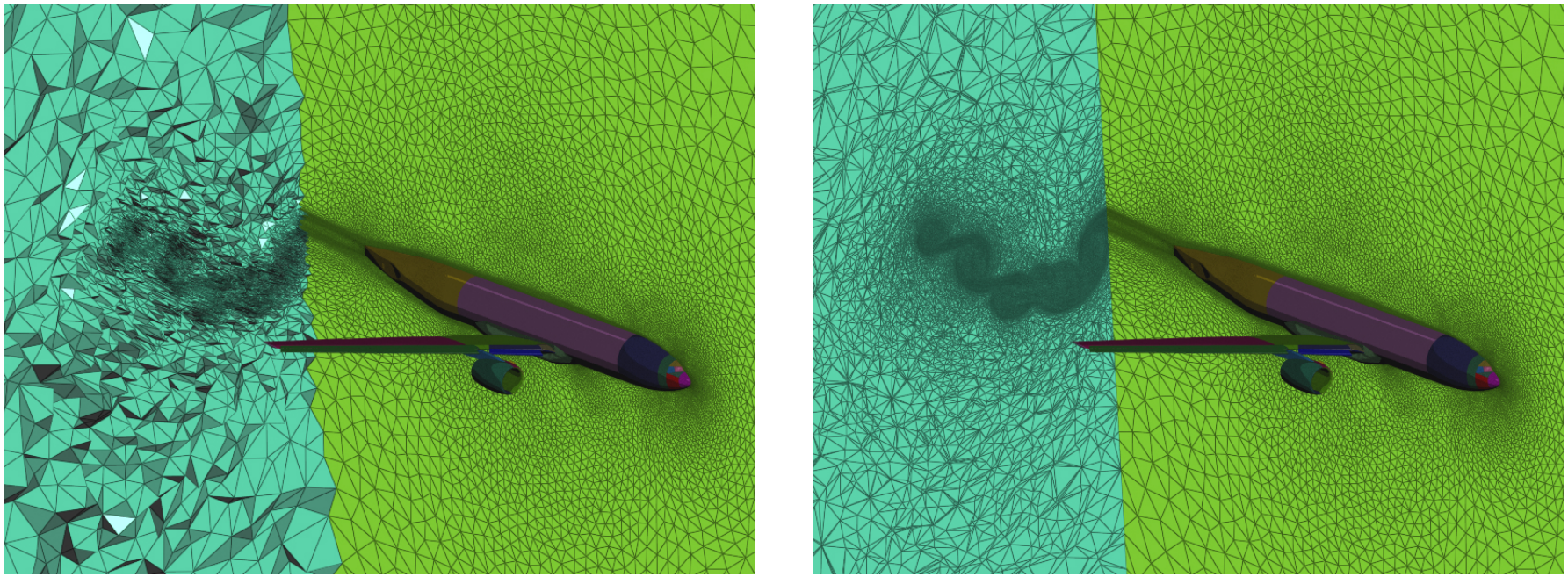
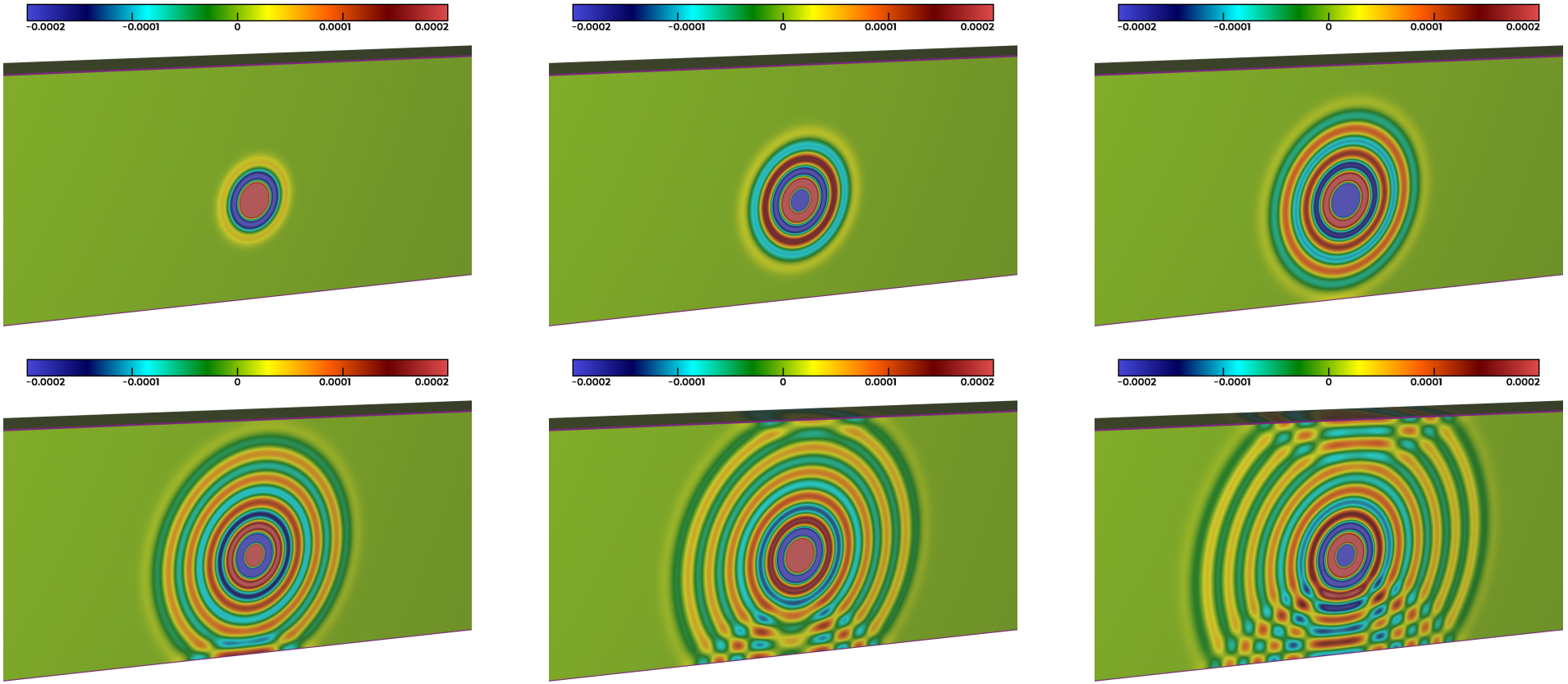


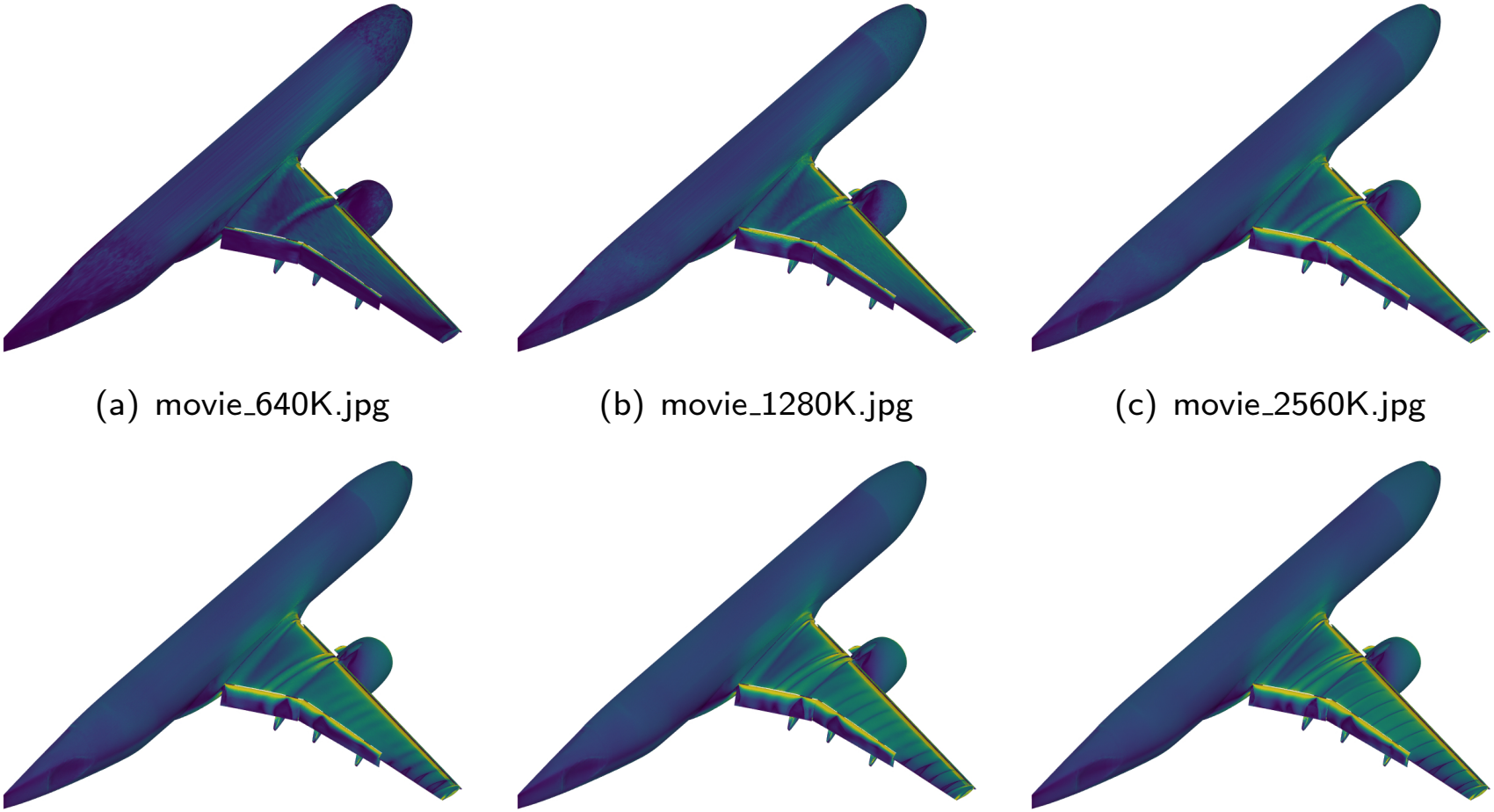
Figure 6: Examples of cut planes.

Scripting tools

Scripting tools to easily generate images or go over large set of meshes / solutions.



Numerical results



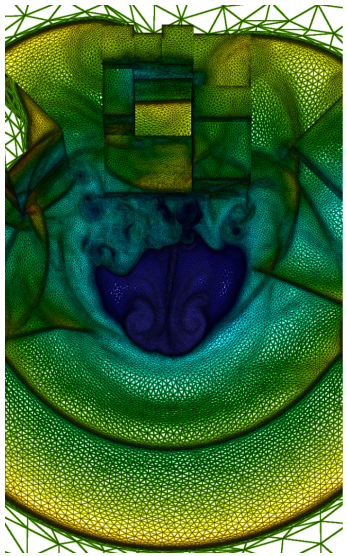
(a) movie_640K.jpg (b) movie_1280K.jpg (c) movie_2560K.jpg

(d) movie_5120K.jpg (e) movie_10240K.jpg (f) movie_20480K.jpg

Figure 7: Friction Coefficient (Cf) solutions for different adapted meshes.

Web site

`ViZiR 4` web site: `https://pyamg.saclay.inria.fr/vizir4.html` with executables (Mac, Linux, Windows), samples (meshes and solutions files) and user guide.



ViZiR 4

ViZiR 4 is a light, simple and interactive high-order meshes and solutions visualization software using OpenGL 4 graphic pipeline.

Its main features are:

- 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.

GO TO VIZIR 4 PAGE FOR MORE DETAILS

VIZIR LEGACY | BACK TO TOP

Download ViZiR

Please follow ViZiR user guide below to have the details on ViZiR installation.

Linux	MacOS	Windows
Download: Download ViZiR for Linux	Download: vizir4.2021.03.03.dmg	Download: vizir4.2021.01.21.windows.zip

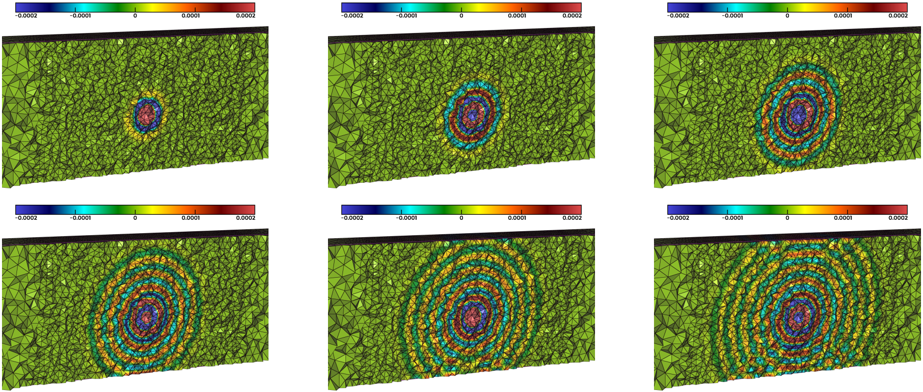
All executables (previous or new versions) can be found in this directory

USER GUIDE | BACK TO TOP

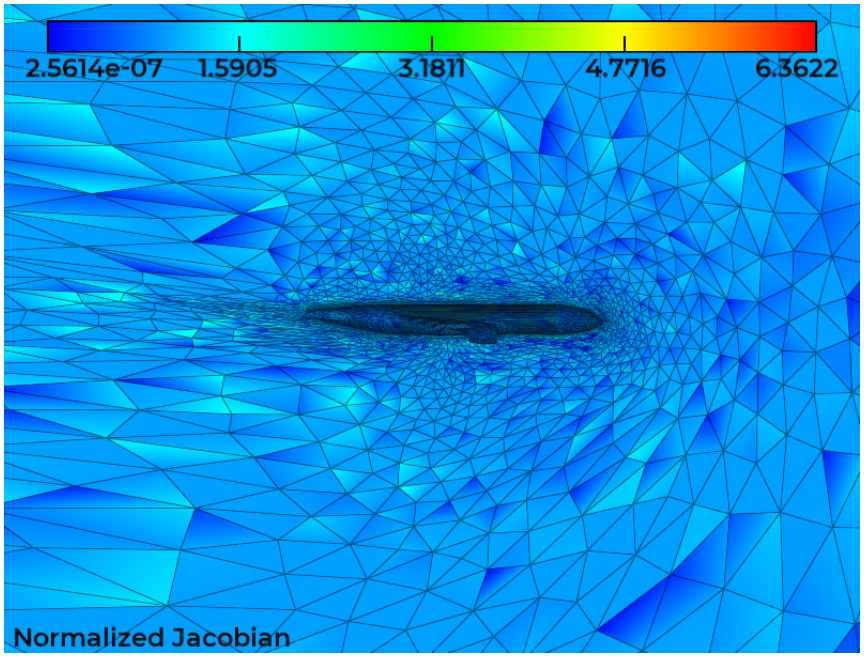
Thank you for your attention

Movie Mode: vizir4 -movie. An example of file vizir.movie

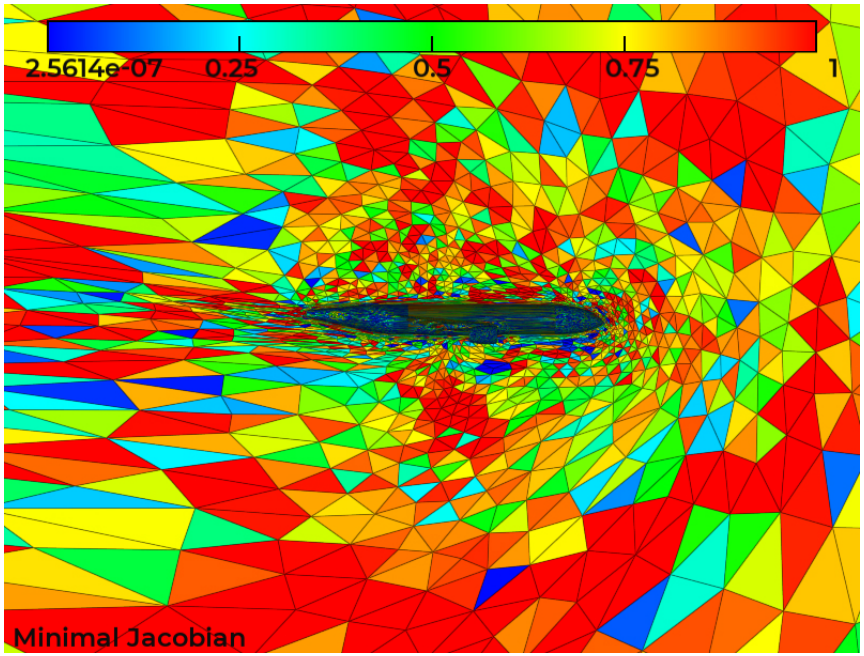
mesh1.meshb sol1.solb
mesh2.meshb sol2.solb
mesh3.meshb sol3.solb
mesh4.meshb sol4.solb
mesh5.meshb sol5.solb
mesh6.meshb sol6.solb



Jacobian



(a) Exact jacobian



(b) Minimal jacobian

Figure 8: Example of jacobian rendering: exact (left) and minimal (right).

Filters

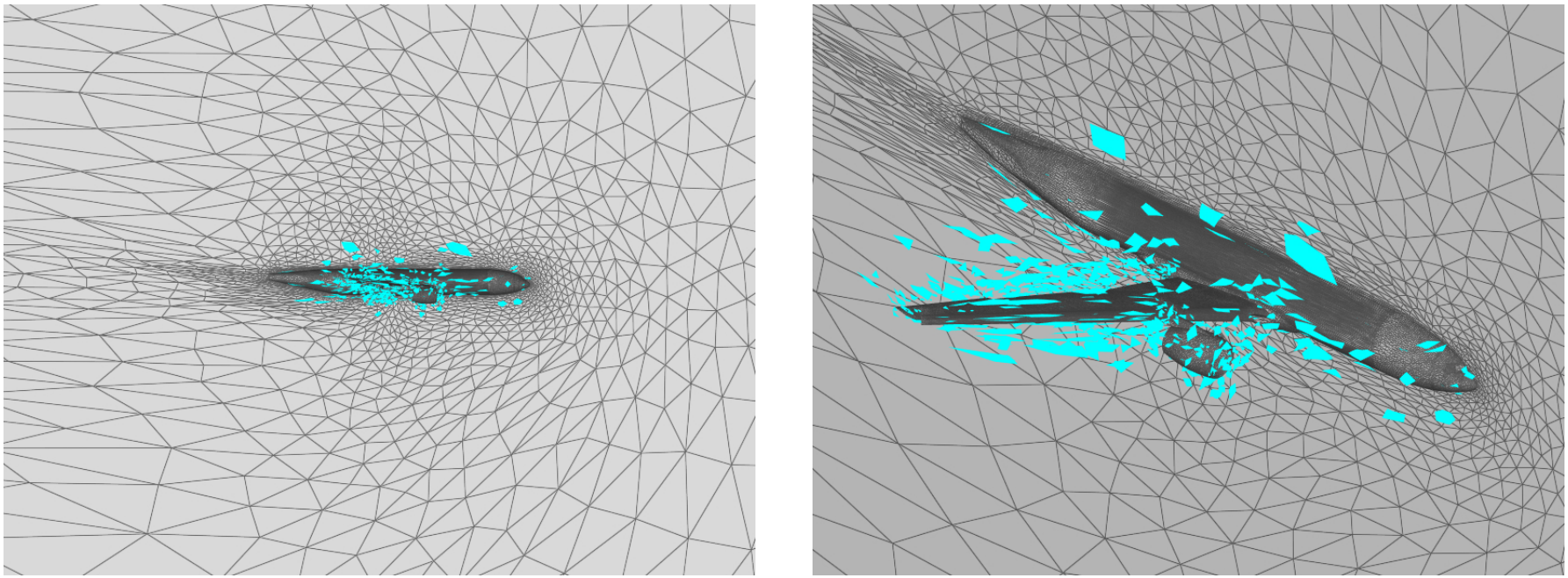


Figure 9: Use of filters: all elements in light blue appear as they belong to the range of the filter (for a given criterion).