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

Visualisation de maillages et solutions de grandes tailles issus de simulations numériques avec ViZiR 4

Matthieu Maunoury, Adrien Loseille

English Abstract—ViZiR 4 is a light, simple and interactive high-order meshes and solutions visualization software using OpenGL 4 graphic pipeline. The use of OpenGL Shading Language (GLSL) allows to perform pixel exact rendering of high order solutions on flat elements and almost pixel exact rendering on curved elements with tessellation done directly on GPU. Post-processing tools, such as picking, isolines, clipping, capping are provided to interact on the fly with the results displayed.

1 Introduction

Numerical simulations are a common way to predict the behavior of physical phenomenons without using prototypes or experimentations. In particular, high-order methods became very popular as they allow to perform complex computations efficiently. Numerical simulations have applications in many fields such as Computational Fluid Dynamic, acoustics, electromagnetism or medical modeling.

The visualization of meshes and solutions is a keystone of the numerical simulations process as it allows to check the validity and quality of the meshes, display the numerical solutions computed and analyze the potential problems on meshes and solutions.

However, the post-processing of high order meshes and solutions is still a current and complex challenge. Indeed, most of the standard visualization softwares (e.g. ParaView [1], Visit [2], Tecplot [12], Gmsh [4]) are based on linear primitives as imposed by the baseline graphic pipeline commonly-used. To bypass these limitations, a low-order remeshing strategy exists. The principle is therefore to define a sub-mesh and affine representations which approximate the solution. A visualization error, corresponding to the gap between the numerical solution and its representation, is therefore introduced and controlled [4], [6], [7], [11], [13] and the rendering obtained is, as a consequence, inaccurate. Some other approaches are based on raycasting [8]–[10]. For each pixel, rays are cast to determine the color for this pixel. However, this solution has limited interactive capabilities [8].

We are developing `ViZiR 4` [3], [5], an interactive and reliable high order meshes and solutions visualization platform, based on OpenGL Shading Language (GLSL).

- *Matthieu Maunoury: GAMMA team, Inria Saclay, France Matthieu.Maunoury@inria.fr.*
- *Adrien Loseille: GAMMA team, Inria Saclay, France E-mail: Adrien.Loseille@inria.fr.*

2 Main features of ViZiR 4

2.1 Fast I/O

A key to have an efficient visualization is to be able to quickly open mesh and solution files. Input and output are handle by the libMeshb[1] library. The files follow the GMF format provided by this library. For instance, the mesh of Lucy (see Fig. 1) with more than 14 millions vertices and 28 millions triangles (642 Mb) is opened is less than 1.5 seconds.

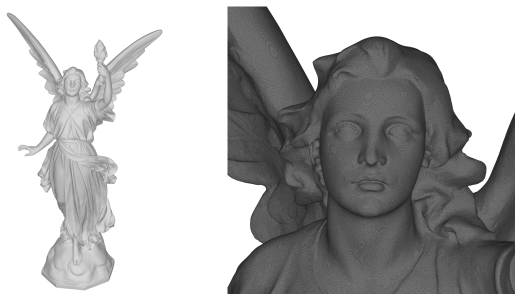

Fig. 1. Rendering of a large mesh of 14M vertices and 28M triangles in 7.5 seconds (total time) on a laptop.

2.2 Pixel exact rendering on flat elements

OpenGL 4 graphic pipeline flexibility allows to compute on the fly the solution. It leads to a pixel exact rendering when flat elements (of degree one) are considered regardless of the degree of the solution. This recent language (GLSL) enables `ViZiR 4` to certify a faithful and interactive depiction. High order solutions are natively handled by `ViZiR 4` on surface and volume (tetrahedra, pyramids, prisms, hexahedra) meshes which can naturally be hybrid. Fig. 2 shows an example of pixel exact rendering of high-order solution.

1. https://github.com/LoicMarechal/libMeshb

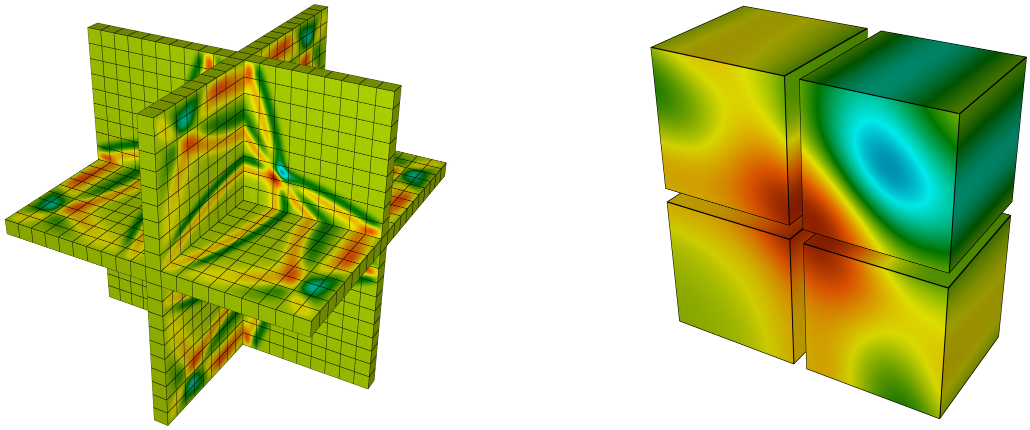

Fig. 2. High-order (degree Q^6) solution of a wave propagation problem. Right: zoom of the solution on 4 hexahedra.

2.3 Tessellation on GPU for high-order elements

When more complex geometries are considered, curved elements perform a better approximation of the geometry. In this case, tessellation shaders occur in OpenGL pipeline (see [3], [5] for more details on the shaders pipeline) to tessellate all elements directly on the GPU. For solutions on such curved elements, almost pixel exact rendering is ensured. An example of curved mesh is shown on Fig. 3 and an example of tessellation is shown on Fig. 4.

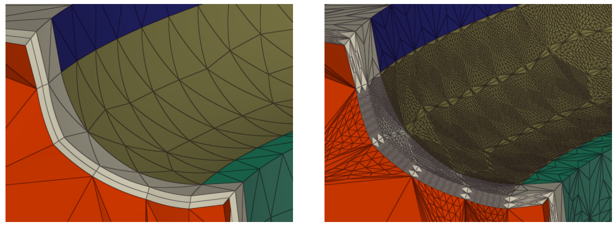

Fig. 4. Rendering of high-order mesh (left) and its tessellation constructed by the GPU (right).

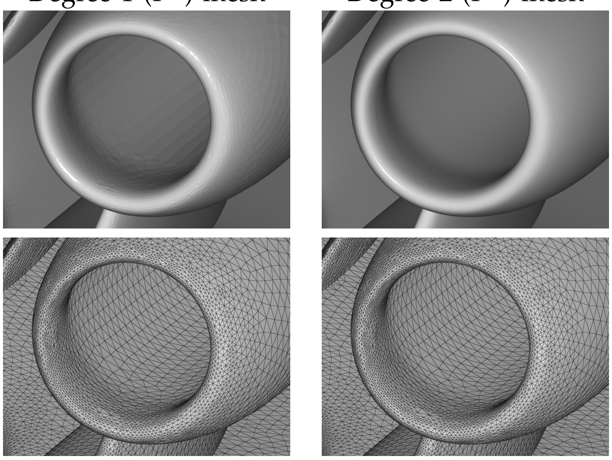

Fig. 3. Comparison of rendering of meshes of degree 1 (left) and 2 (right) for the same number of elements.

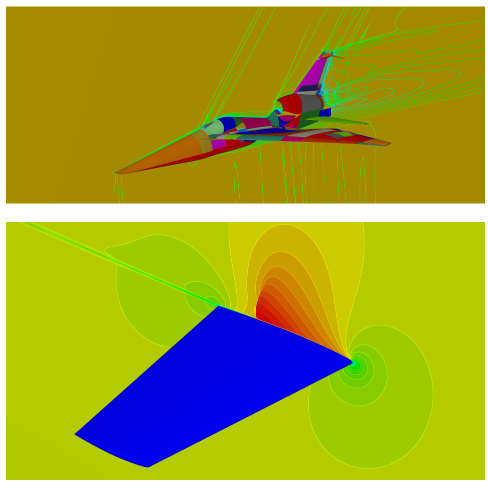

Fig. 5. Examples of isolines rendering.

2.4 Post-processing tools and interactivity

Many post-processing tools are available to make the analyses of the results possible. For instance:

- Picking elements to get information.
- Hide surfaces by reference (after picking).
- Isolines rendering (see Fig. 5).
- Clip planes. All volume elements belonging to a plane are displayed (see Fig. 6).
- Filters. According to a criterion, for instance element quality or minimal jacobian, all elements in a given range of values are filtered and displayed in a different color than others (see Fig. 7).

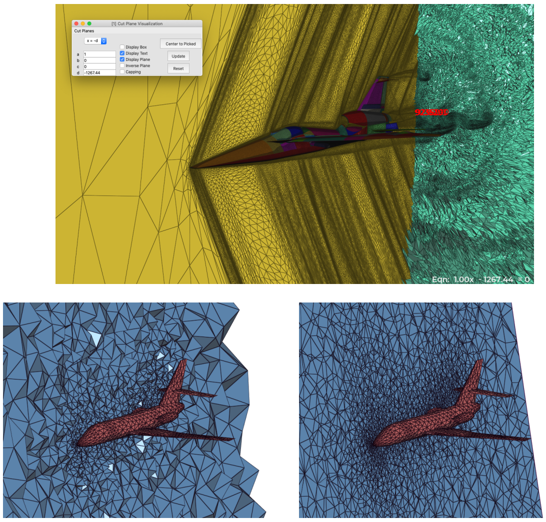

Fig. 6. Examples of cut planes.

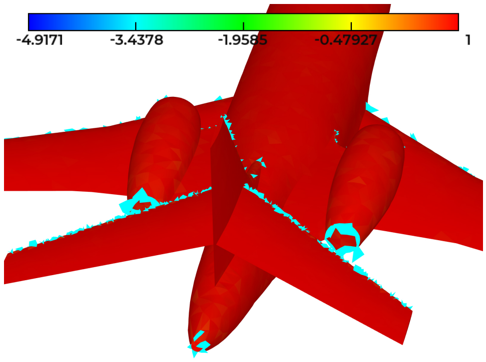

Fig. 7. Use of filters. All elements in light blue appear as they belong to the range of the filter (for a given criterion).

2.5 Rendering performances

All the results collected have been generated with the same laptop: a MacBook Pro with 2.6 GHz 6-core Intel Core i7 with 32 Gb of RAM, and the GPU is a AMD Radeon Pro Vega 20 4 Gb. For the large mesh of Lucy (see Fig. 1) with more than 14 millions vertices and 28 millions triangles (642 Mb), the number of Frames Per Seconds (FPS) is around 28, which is enough to be interactive. All other results are extremely fluid with 60 FPS (requested maximum).

3 Conclusion

More details on `ViZiR 4`, some examples, samples and executables can be found on our website http://vizir.inria.fr

References

[1] U. Ayachit. *The paraview guide: a parallel visualization application*. Kitware, Inc., 2015.
[2] H. Childs. Visit: An end-user tool for visualizing and analyzing very large data. 2012.
[3] R. Feuillet, M. Maunoury, and A. Loseille. On pixel-exact rendering for high-order mesh and solution. *Journal of Computational Physics*, 424:109860, 2021.
[4] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering*, 79(11):1309–1331, 2009.
[5] 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.
[6] M. Maunoury. *Méthode de visualisation adaptée aux simulations d'ordre élevé. Application à la compression-reconstruction de champs rayonnés pour des ondes harmoniques.* PhD thesis, 2019.
[7] M. Maunoury, C. Besse, V. Mouysset, S. Pernet, and P.-A. Haas. Well-suited and adaptive post-processing for the visualization of hp simulation results. *Journal of Computational Physics*, 375:1179–1204, 2018.
[8] B. Nelson, E. Liu, R. M. Kirby, and R. Haimes. Elvis: A system for the accurate and interactive visualization of high-order finite element solutions. *IEEE transactions on visualization and computer graphics*, 18(12):2325–2334, 2012.
[9] B. W. Nelson. *Accurate and interactive visualization of high-order finite element fields*. PhD thesis, 2012.
[10] J. Peiro, D. Moxey, B. Jordi, S. Sherwin, B. Nelson, R. Kirby, and R. Haimes. High-order visualization with elvis. In *IDIHOM: Industrialization of High-Order Methods-A Top-Down Approach*, pages 521–534. Springer, 2015.
[11] W. J. Schroeder, F. Bertel, M. Malaterre, D. Thompson, P. P. Pebay, R. O'Bara, and S. Tendulkar. Methods and framework for visualizing higher-order finite elements. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):446–460, 2006.
[12] TecPlot Inc. TecPlot. https://www.tecplot.com/.
[13] L. Xu, X. Ren, X. Xu, H. Li, Y. Tang, and Y. Feng. An adaptive visualization tool for high order discontinuous galerkin method with quadratic elements. In *2017 IEEE International Conference on Computer and Information Technology (CIT)*, pages 176–183. IEEE, 2017.