6
Conclusion

We have presented an accurate hexahedral volume rendering suitable for unstructured meshes. Our proposal integrates the trilinear scalar function using a quadrature approach on the GPU. Although our performance was in average 12% worse than a tetrahedral algorithm, our proposal produces images with better quality. By the use of a proper integration of the trilinear function inside an hexahedron, our proposal ensures that no isosurface value is missed during integration. We also presented a fast and high-quality algorithm for hexahedral meshes that, although does not achieve images as accurate as the ones using quadrature integration, achieves good approximations and is also faster than a tetrahedral volume rendering.

Because of its parallel nature, we believe that ray-casting is better suited for massively parallel environments, such as the GPU. Its main drawback, however, is its memory consumption; our algorithm presents a smaller memory footprint than regular hexahedral subdivision schemes.

In the future, we plan to study the impact of a ray-bilinear patch intersection test, and to analyze how it could improve the image quality and its impact on the performance. We also plan to extend the algorithm to higher-order cells and to investigate its use for rendering models that support adaptive level-of-detail.