Immersive and Interactive 3D Visualization of Large-Scale Geo-Scientific Data

Markus Flatken *
German Aerospace Center (DLR)

Simon Schneegans †
University of Bremen

Riccardo Fellegara ‡
German Aerospace Center (DLR)

Andreas Gerndt §
German Aerospace Center (DLR)
University of Bremen

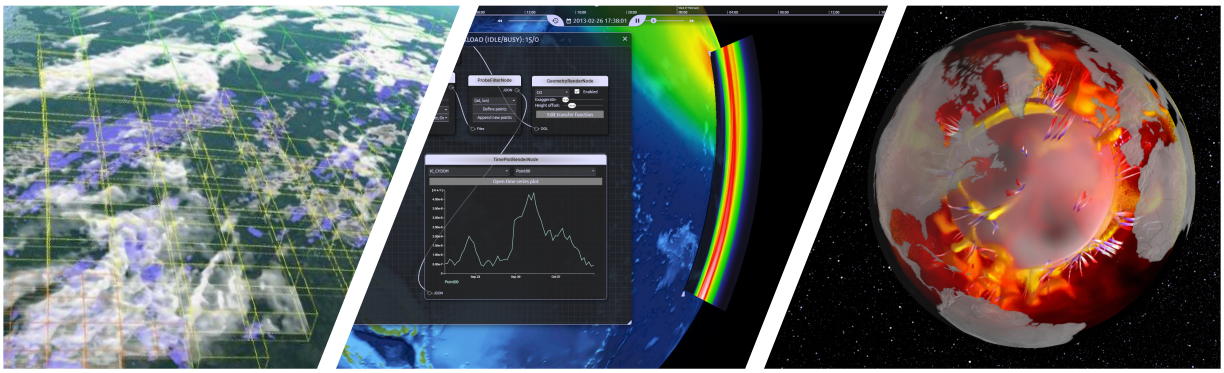

Figure 1: The images above were all produced using the framework described. The first image displays a pseudo-volumetric rendering of a 2.6 TB weather simulation, in which interactive exploration of the time-dependent data set is enabled through view-dependent, distributed feature extraction. The framework also enables exploration of simulation data stored on remote clusters, such as large-scale climate simulations, as shown in the center image. The last example does not use a distributed data processing backend but the highly parallel ray tracer OSPRay developed by Intel.

Abstract

In this paper, we present a software architecture and framework developed over the past decade to enable scalable and highly interactive visualizations for large datasets and display sizes. The framework integrates distributed data processing, data streaming, and dynamic scheduling to allow for view-dependent feature extraction and progressive data streaming. Additionally, the framework has been designed to support visualizations from local desktop workstations to large multi-display virtual environments.

Index Terms: Computing methodologies—Modeling and simulation—Simulation types and techniques—Scientific visualization; Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality

1 Introduction

The visualization of large-scale scientific datasets is a demanding task. Local resources available on a single workstation do not provide the necessary compute capability to analyze and visualize such datasets interactively, where the user continuously changes the parameters and styling of the visualization itself. In such scenarios, the response time, i.e., the time until changes are visible to the user, is often unacceptably high, hindering interactive and explorative data analysis.

Hence, there is a need to distribute the computation, which is why distributed client/server applications like ParaView [1] or VisIt [2] have been developed. These applications offload heavy processing to a server running on a High-Performance Computing (HPC) cluster, while allowing the user to interact with a locally running graphical user interface. This distribution drastically improves response times, but in some applications, such as Virtual Reality (VR) setups where interaction with the data is crucial, the response times are not low enough and the applications often still stutter or freeze until results are available. Progressive data streaming is a possible solution, but these techniques are only rarely available. Additionally, the data distribution for parallel processing typically follows a static scheduling approach, where data is distributed to processing nodes based on its spatiality. This can result in some processes experiencing high load during computation, while others remain idle during a query.

We presume that there is a lack of software frameworks which allow for parallel processing of large-scale scientific datasets, data streaming, as well as parallel rendering of the extracted data in multi-display virtual environments. This overall prohibits the scaling of such applications from the desktop, to local head-mounted displays (HMD), up to huge VR installations. Consequently, domain scientists seldomly have the chance to explore their data in large-scale virtual environments.

In this paper, we introduce a software architecture and framework that have been developed over the past decade to enable scalable and highly interactive visualizations for large datasets and display sizes. In order to manage these large-scale datasets, we have integrated distributed data processing, data streaming, and dynamic scheduling, where data is distributed not only based on its spatiality but also considering additional meta-information such as active scalar ranges, or the current camera perspective. This allows for view-dependent feature extraction and progressive data streaming. Additionally, we have developed a scalable visualization frontend that supports visualizations from local desktop workstations optionally equipped with a head-mounted display up to large multi-display virtual environments.

*e-mail: markus.flatken@dlr.de
†e-mail: sschneeg@uni-bremen.de
‡e-mail: riccardo.fellegara@dlr.de
§e-mail: andreas.gerndt@dlr.de

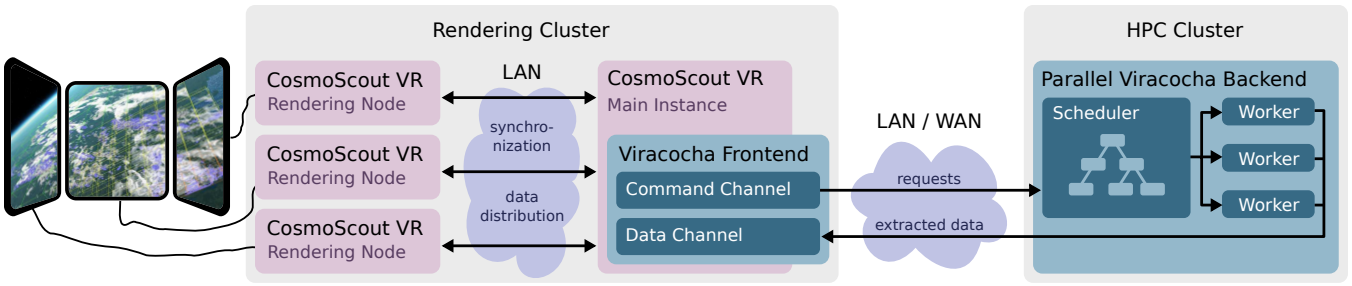

Figure 2: Viracocha is integrated as a CosmoScout VR plugin to enable client/server based analysis and visualization of large-scale datasets. The heavy workload, such as feature extraction, is outsourced to a parallel backend application running on remote resources, such as a HPC cluster. Compute requests are sent to the backend where they are distributed to a set of worker processes. The extracted features are finally streamed to the main instance of CosmoScout VR or directly to the rendering nodes. The frontend can also run on a cluster of rendering nodes, in order to drive multi-pipe virtual environments or tiled displays.

The structure of this paper is as follows: In the next section, we introduce the software architecture and provide a brief overview of the different domains where the software is being utilized. Then, we describe the various hardware systems where this system can be used. We also provide exemplary performance measurements to demonstrate the interactivity of the system. Lastly, we conclude with some lessons learned and an outlook on future developments.

2 Software Architecture

Our software framework is composed of both a frontend and backend application. The frontend application provides interactive 3D visualization and can be configured to run on a single workstation or on a cluster of multiple machines to power stereoscopic multi-display setups. The backend can be configured to run on the same machine as the frontend or on the compute nodes of a HPC cluster, allowing for distributed computation and data processing. Figure 2 shows a high-level overview of the involved components.

2.1 The Frontend: CosmoScout VR

As visualization frontend, we developed CosmoScout VR, an open source 3D solar system [11]. The software enables scientists to virtually navigate from planet to planet and explore time-dependent data across many magnitudes of scales. A key feature of CosmoScout VR is its precise rendering of planet-scale terrain datasets (elevation and imagery) using standard web-map service protocol (WMS). With CosmoScout VR as frontend application, we can visualize many different kinds of datasets, as long as they have some kind of spatial reference. Examples include, but are not limited to, climate or weather simulation data, satellite data, subsurface data, or interplanetary data such as space-weather data.

CosmoScout VR supports Virtual Reality devices such as head-mounted displays (HMDs) or stereoscopic multi-projection systems like CAVEs, but can also be used on traditional desktop PCs. In fact, the user interface of CosmoScout VR has been developed in a desktop-first fashion. We understand that domain scientists do not typically work in Virtual Reality, but rather on their desktop workstations. Consequently, we focus on developing interactive tools that are primarily designed for desktop usage, but also scale well to VR setups.

Most of the functionalities of CosmoScout VR are loaded at runtime from plugins. This allows for fast prototyping development times on the one hand; on the other hand, different features can be developed independently, as the core modules usually do not need to be changed. To support the analysis of large-scale geo-scientific datasets using remote HPC resources, CosmoScout VR has been extended with a plugin providing a graph-based user interface (visible in the center image of Figure 1) and local rendering algorithms for meshes and volumes. The graph-based user interface enables scientists to configure and control the overall data analysis pipelines, executed on the backend, or modify parameters of the rendering algorithms. Since CosmoScout VR uses HTML and JavaScript to render the 2D user interface elements, this node graph could be implemented using the D3.js framework.

2.2 The Backend: Viracocha

The results of large-scale numerical simulations are usually not stored on a local hard drive. These datasets are mainly generated by using parallel solvers running on HPC resources. Copying the generated raw data, often multiple terabytes, from the HPC cluster to the local workstation for analysis is impracticable or even impossible. In order to avoid this data transfer, we opted for a client/server architecture.

The server application has been developed based on the Viracocha middleware layer [6] using MPI [3]. It is executed on a HPC system, and allows for efficient feature extraction. Examples include the extraction of vertical profiles or probing a specific location of a time-dependent climate simulation (depicted in the center image of Figure 1), or the clouds of a weather simulation represented by different iso-contours [10] as depicted on the left in Figure 1.

When a user requests the extraction of a specific feature, a message is sent over the network to the parallel backend application using the command channel. When received, the scheduler creates independent tasks which are scheduled for parallel processing using multiple workers. These workers finally execute the feature extraction algorithms on the data and results are immediately streamed back to CosmoScout VR (see Figure 2). When focusing on interactive data exploration, this is an iterative process which leads to costly data processing and heavy I/O demands. Parallelizing feature extraction improves response times — the time a user has to wait until results are visible — but when data is really huge, execution times are still high. In order to further increase the interactivity, optimized scheduling strategies are employed. These strategies define the order of data processing on the worker nodes, and can utilize tree-like accelerations structures for view-dependent data processing and progressive data streaming as presented in [4, 10].

In order to enable the development of domain specific solutions, programming interfaces are offered as often as possible. These interfaces enable application developers to integrate own algorithms, messages types, scheduling strategies, and custom user interfaces into Viracocha and CosmoScout VR.

3 Use Cases

The following subsections describe some examples of domain specific applications in which we use the described system to enable interactive data analysis and visualization.

3.1 Exploration of Weather Simulations

Weather simulations produce large time-dependent data which is difficult to analyze interactively. An example dataset was provided for the 2017 IEEE SciVis Contest by the HD(CP)² project [10]. This dataset contains 240 time steps with a total of 2.6 TB. Each timestep contains the weather conditions over Germany in a 3D volume of multiple chemical scalars.

In order to interactively explore this data, we have decomposed each time step into small chunks of data which are used as leaf nodes in a multi-resolution octree. Viracocha is then used to extract features in parallel, giving a higher priority to octree nodes close to the virtual camera.

Hence, when the user moves through the data or changes a parameter, e.g. a threshold of a scalar range, results are progressively generated by the Viracocha workers and immediately streamed to CosmoScout VR, always updating parts in the user's vicinity first. The same happens when the simlation time progresses: parts of the data closest to the user will reflect the change in a matter of milliseconds, while data in the distance may take a couple of seconds to be updated. A screenshot presenting this use case is provided in the first image of Figure 1. Additionally, we present exemplary performance evaluations of this application in Section 5.

3.2 Exploration of Chemistry-Climate Simulations

Within the "Earth System Chemistry Integrated Modelling (ESCiMo)" initiative, chemistry-climate-simulations were executed to support upcoming scientific assessments by the World Meteorological Organization (WMO), the United Nations Environment Programme (UNEP), and the Intergovernmental Panel on Climate Change (IPCC) [7]. The analysis of the resulting data represents an important and yet difficult task in global climate change research. It is challenging due to the multi-dimensional data including hundreds of chemical variables, a high temporal resolution covering nearly 100 years of simulation time, as well as due to the sheer data size of more than one petabyte.

To enable interactive analysis of this data, we added multiple filters to Viracocha, such as the parallel computation of averages and standard deviations for time ranges, the extraction of vertical profiles using 3D fields, or probing and plotting of time series values at user defined positions. Additionally, visualizing differences for two of these filters enables comparing different simulations of the overall ensemble. Another common use-case is to map chemical values onto isosurfaces extracted on different scalars. An example of such a setup is shown in the second image of Figure 1.

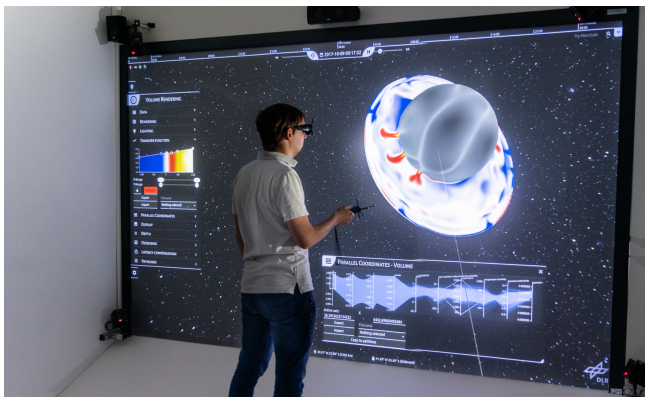

Figure 3: The desktop-oriented user interface of CosmoScout VR is also used in Virtual Reality. While the interaction with complex widgets like a transfer function editor or a parallel coordinates plot is not ideal, this eases the learning process for domain scientists.

Figure 4: The Virtual Reality lab at the German Aerospace Center in Brunswick, Germany is used regularly to evaluate the framework described in this paper. The lab uses four stereo projectors (three for the back projection and one for the floor projection). Each projector is connected to a separate workstation. Images courtesy of German Aerospace Center, DLR (CC-BY).

3.3 Visualization of Mantle Convection

While enormous data collected from space missions and telescopes enhanced the understanding of the thermo-chemical evolution of terrestrial planets, the understanding of their interior dynamics is still limited. However, over the last decades, numerical simulations of the planet interiors have become powerful tools to model the evolution of mantle flow [8].

In order to analyze the resulting data together with a high-resolution terrain topography, Viracocha has been extended with appropriate filters for contouring and slicing. Most filters support time-parallelization, where multiple time steps are processed concurrently on the Viracocha workers. Additionally, an interactive volume-rendering approach has been developed [5]. The corresponding plugin for CosmoScout VR is based on Intel's OSPRay Raytracing toolkit. In this case, the data processing and rendering is done locally. Yet the ray-tracing of the volume runs asynchronously to not affect the interactivity of the entire frontend application. To hide the vayring frame rates, we integrated image warping techniques. These were also used to generate stereoscopic image pairs when running in a VR setup. An example of this volume rendering is shown in the last image of Figure 1 as well as in Figure 3.

4 Hardware

The framework is mostly being developed on desktop workstations, however it is frequently tested and evaluated not only on HMDs, but also in the Virtual Reality lab at the German Aerospace Center in Brunswick, Germany (shown in Figure 4). Here, we also have access to a high-performance data analytics (HPDA) cluster for running the backend application.

The Virtual Reality lab, in which we use the software on a daily basis, features a back projection, a floor projection, active stereo, and head tracking. The displays are powered by four Barco F50 projectors. Three projectors are mounted in a backroom to drive the powerwall and one is mounted on the ceiling for the floor projection. The powerwall uses three projectors in order to increase the pixel resolution. Each projector is connected to a workstation equipped with 2x Intel Xeon E5-2630 v3 CPUs, 128 GB of main memory, and a single NVIDIA Quadro P6000 graphics card.

For evaluation purposes, the backend is usually executed as a parallel application on a HPDA cluster comprised of four compute nodes. Each node is equipped with four Intel Xeon Skylake 6132 processors and has 384 GB of main memory. Each CPU of each node runs at 2.60 GHz and has 14 cores. The cluster is connected with 4 Gbit/s to the VR environment shown in Figure 4.

5 Performance Example

In order to benchmark the performance of our software framework when analyzing large scale datasets, we executed CosmoScout VR in combination with a Viracocha backend executed on the HPDA cluster. The weather simulation data and plugins used are from the 2017 IEEE visualization contest [10].

Figure 5 depicts the measured timings for parallel and view-dependent extraction of four isosurfaces. Timings are captured starting from a low resolution perspective, where surfaces are generated for just a few octree nodes. When zooming in, the resolution increases and a continuous stream of requests is sent to the backend (orange line). Most of the time, there are enough workers available to instantly process the requests and send back the extracted geometry of the isosurfaces. The amount of rendered octree nodes (green dotted line) steadily increases up to 1103. At frame 288, the simulation time step is changed, which leads to a complete re-processing of all visible nodes. As requests are prioritized by view distance, the octree nodes close to the camera are updated first. Initial results are available after about 80 ms. And after 2.4 s, the complete scene is updated. One time step of the octree contains about 12.6 GB of data from which 658.4 MB of geometry have been extracted and send to the CosmoScout VR client for the benchmarked perspective.

The table below depicts the achieved timings when launching the backend with a varying number of MPI processes.

Backend Processes	8	16	32	64	128	256
Update Time	25.0 s	12.7 s	6.7 s	3.9 s	2.4 s	2.3 s

The time for re-processing all 1103 visible nodes with 8 MPI processes took 25 s. When running with 256 MPI processes, the time could be reduced to 2.3 s. We can see that up to 128 processes we have nearly linear scaling. However, when the number of processes further increases, the efficiency decreases. This is both due to the limited network bandwidth available between the CosmoScout VR clients and the HPDA cluster, and the required disc reads of the raw data. Nevertheless, during the entire session the overall frame time stayed mostly below 16.67 ms which allow us to explore the 2.6 TB dataset interactively.

6 Conclusion and some Lessons Learned

In this paper, we presented the architecture of a software framework which can be used to visualize and explore large-scale geo-scientific data. We combined CosmoScout VR and Viracocha in order to create a system which can be scaled in terms of rendering clients for the frontend as well as compute nodes for the backend. Hence, it allows creating applications which can be launched on a single workstation as well as on a distributed rendering cluster connected to an HPC cluster.

For our software development, we chose a desktop-first policy. We understand that domain scientists do not typically work in Virtual Reality but rather at their local desktop workstations. Hence, we focus on developing interactive tools that are primarily designed for desktop usage but also scale well to VR setups. This is supposed to reduce the barrier for domain scientists to explore and present their data in virtual environments. We think that this approach is not only valuable for data analysis but can also foster effective communication of scientific findings to colleagues and to the general public, since the same software can be used for data analysis as well as for data presentation.

During development, we found that device input, interaction with the virtual scene, and 3D navigation are areas where it has been fairly simple to map the desktop interactions to Virtual Reality input devices. However, the main challenge we have faced is in the user interface. A traditional 2D interface with buttons, sliders, and text input fields can look out-of-place in VR and can be difficult to

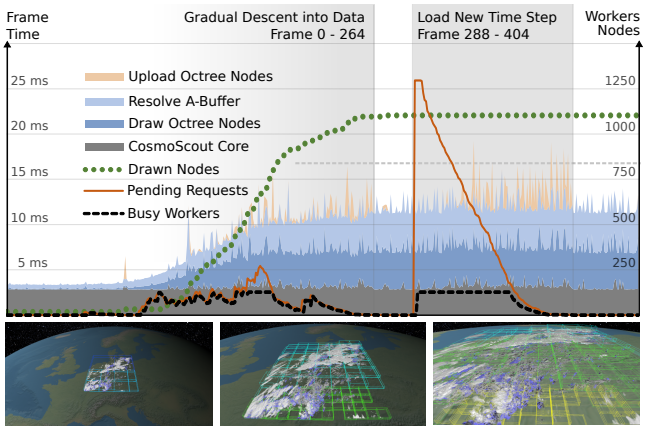

Figure 5: In the first half of the 450 frames shown in the graph above, the camera approached the weather data located above Germany. At frame 288, the visualized time step was changed leading to many pending requests. The graph shows GPU timings as well as the number of rendered octree nodes, pending compute requests and currently active worker processes on the parallel backend. The dotted grey line indicates a frame time of 16.67 ms.

interact with. Especially complex user interfaces like the dataflow graph editor do not work well in Virtual Reality. Nevertheless, we have chosen to show the entire 2D interface in VR. This approach eases the transition for users as there is no new interface to learn. Furthermore, it greatly reduces development overhead since it does not require developing two user interfaces in parallel.

Also, the decision to put most of the functionality of the CosmoScout VR frontend into plugin libraries has paid of very well. On the one-hand side, this reduces development time significantly as a recompiled plugin can be hot-reloaded at runtime without the need to restart the application. On the other hand, it greatly simplifies the maintainability of the software in a research context, where parallel development of long-living branches is common. Usually, different plugins are developed in different projects and even after months of development it will be possible to merge them into the main development branch if necessary.

As the software can be deployed on various hardware setups, we can compare analyzing the same data on different hardware setups. Overall, we see three main advantages of Virtual Reality laboratories when compared to head-mounted displays. First, a large VR installation typically induces less motion sickness and users can work for longer periods than when working with an HMD. Second, a VR laboratory is an impressive installation which we frequently use for presenting to visitors such as the general public. This would not be possible with a single HMD. Third, a laboratory can provide much more computing resources to power the display setup. When visualizing large datasets, both performance and display resolution are crucial. While there are high-resolution HMDs available, it is very difficult to achieve high frame rates as they are only connected to a single GPU. With a multi-display setup, we can scale the resolution by adding additional rendering nodes.

In the future, we plan to release Viracocha as an open-source software. The presented frontend application — CosmoScout VR — is already open source [9]. Furthermore, we plan to revisit contemporary game engines such as Unity or Unreal and evaluate whether they could serve as an alternative frontend for our framework. Over the years, the software has matured significantly and as it became more and more stable it now serves as a solid basis, both for research regarding desktop-based scientific data visualization techniques, and for new Virtual Reality approaches.

REFERENCES

[1] J. Ahrens, B. Geveci, and C. Law. ParaView: An end-user tool for large data visualization. *Visualization Handbook*, 2005.

[2] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rübel, M. Durant, J. M. Favre, and P. Navrátil. VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data. In *High Performance Visualization–Enabling Extreme-Scale Scientific Insight*, pp. 357–372. Oct 2012.

[3] L. Clarke, I. Glendinning, and R. Hempel. The MPI message passing interface standard. Technical report, 1994.

[4] M. Flatken, A. Berres, J. Merkel, I. Hotz, A. Gerndt, and H. Hagen. Dynamic Scheduling for Progressive Large-Scale Visualization. In E. Bertini, J. Kennedy, and E. Puppo, eds., *Eurographics Conference on Visualization (EuroVis) - Short Papers*. The Eurographics Association, 2015. doi: 10.2312/eurovisshort.20151122

[5] J. Fritsch, M. Flatken, S. Schneegans, A. Gerndt, A.-C. Plesa, and C. Hüttig. Raypc: Interactive ray tracing meets parallel coordinates, 2022. doi: 10.48550/ARXIV.2207.12011

[6] A. Gerndt, B. Hentschel, M. Wolter, T. Kuhlen, and C. Bischof. Viracocha: An efficient parallelization framework for large-scale CFD post-processing in virtual environments. pp. 50– 50, 12 2004. doi: 10.1109/SC.2004.66

[7] P. Jöckel, H. Tost, A. Pozzer, M. Kunze, O. Kirner, C. A. M. Brenninkmeijer, S. Brinkop, D. S. Cai, C. Dyroff, J. Eckstein, F. Frank, H. Garny, K.-D. Gottschaldt, P. Graf, V. Grewe, A. Kerkweg, B. Kern, S. Matthes, M. Mertens, S. Meul, M. Neumaier, M. Nützel, S. Oberländer-Hayn, R. Ruhnke, T. Runde, R. Sander, D. Scharffe, and A. Zahn. Earth system chemistry integrated modelling (ESCiMo) with the modular earth submodel system (MESSy) version 2.51. *Geoscientific Model Development*, 9(3):1153–1200, 2016. doi: 10.5194/gmd-9-1153-2016

[8] A.-C. Plesa and C. Hüttig. Numerical simulation of planetary interiors: Mantle convection in a 2D spherical shell. In *Workshop on Geodynamics*, 2008.

[9] S. Schneegans, M. Flatken, and A. Gerndt. CosmoScout VR. doi: 10.5281/zenodo.3381953

[10] S. Schneegans, L. Neary, M. Flatken, and A. Gerndt. STRIELAD - a scalable toolkit for real-time interactive exploration of large atmospheric datasets. In *IEEE Visualization*, Oktober 2017.

[11] S. Schneegans, M. Zeumer, J. Gilg, and A. Gerndt. CosmoScout VR: A Modular 3D Solar System Based on SPICE. In *2022 IEEE Aerospace Conference (AERO)*, pp. 1–13. IEEE, 2022. doi: 10.1109/AERO53065.2022.9843488

[12] N. Q. Zhu. *Data visualization with D3. js cookbook*. Packt Publishing Ltd, 2013.