SpaceOps-2023, ID #426

Explorative, Immersive Visualization of Space Weather Phenomena

Riccardo Fellegara[a*], Katharina Krösl[b], Stefano Markidis[c], Frank-Cyrus Roshani[d], Markus Flatken[a], Jonathan D. Fritsch[a], Andreas Gerndt[a,e], Anton Fuhrmann[b], Matthias Hürbe[b], Daniela Stoll[b], Johannes Klug[f], and Evridiki V. Ntagiou[f]

[a] *Department of Software for Space Systems and Interactive Visualization, Institute for Software Technology, German Aerospace Center (DLR), Lilienthalplatz 7, 38108, Braunschweig, Germany, Riccardo.Fellegara@dlr.de, Markus.Flatken@dlr.de, Jonathan.Fritsch@dlr.de, Andreas.Gerndt@dlr.de*
[b] *VRVis Zentrum für Virtual Reality und Visualisierung, Donau-City-Strasse 11, 1220, Wien, Austria, kroesl@vrvis.at, fuhrmann@vrvis.at, huerbe@vrvis.at, stoll@vrvis.at*
[c] *KTH Royal Institute of Technology in Stockholm (KTH), Brinnellvagen 8, 100 44, Stockholm, Sweden, markidis@kth.se*
[d] *Space Operations and Astronaut Training, German Aerospace Center (DLR), Münchener Straße 20, 82234, Weßling, Germany, Frank-Cyrus.Roshani@dlr.de*
[e] *Center for Industrial Mathematics (ZeTeM) at the University of Bremen, Bibliothekstraße 1, 28359, Bremen, Germany gerndt@uni-bremen.de*
[f] *Applications and Robotics Data Systems Section, Mission Operations Data Systems Division, European Space Operations Centre, Robert-Bosch Str., 64293, Darmstadt, evridiki.ntagiou@esa.int, johannes.klug@esa.int*
* Corresponding Author

Abstract

Interactive data exploration represents a key challenge for both the analysis and visualization pipelines in various domains, due to the exponential growth of the data generated by modern sensors. Thanks to recent advances in high-performance computing, processing large-scale data becomes feasible, but the visualization and interactive exploration of such data remain an open challenge. While exploring multi-dimensional data on two-dimensional displays has been a research topic for decades, technologies such as virtual reality (VR) or augmented reality (AR) now allow us to leverage the third dimension and explore three-dimensional data in its spatial context, using immersive analytics, which is seen as an emerging research field and key to support analytical reasoning and decision making. Some modern visualization tools are already exploiting VR and AR to enhance data visualization and exploration, but many existing solutions are limited in functionality, immersivity, or usability. There are several challenges to overcome when developing immersive experiences. In this paper, we present an approach for explorative, immersive visualization of space weather phenomena. The main goal of this work is to define a pipeline that starts from the output of a space weather simulation (e.g., coming from ENLIL or EUHFORIA models), and executes a data preparation step in which the data is reduced to the specific needs of a given visualization tool. This post-processed data is visualized by two distinct visualization tools: one targeting domain experts, and the latter targeting the general public. The final objective of this activity is to provide tools for helping experts interpret complex space weather phenomena, and raising awareness for space weather conditions and their effects on our life on Earth, in the general public.

Keywords: Space Weather, Virtual Reality, Coronal Mass Ejection

Acronyms/Abbreviations

AR	Augmented Reality
CME	Coronal Mass Ejection
ESA	European Space Agency
EUHFORIA	European Heliospheric Forecasting Information Asset
FOV	Field of View
GONG	Global Oscillation Network Group
HEEQ	Heliocentric Earth Equatorial
HMD	Head-Mounted Displays
LOD	Level-of-Detail
MHD	Magneto Hydro Dynamics

NASA	National Aeronautics and Space Administration
PFFS	Potential Field Source Surface
SWE	Space Weather
VR	Virtual Reality
VTK	Visualization ToolKit

Nomenclature

AU	Astronomical Unit is the distance from the Sun to the Earth. One AU equals 149,597,870,691 m.
Rs	Solar Radius is the distance from the center of the Sun to its surface, 696,000 kilometers.

1. Introduction

Despite the advances in analysis methods, data visualization plays a crucial role in understanding the outputs of space weather models. Space weather (SWE) phenomena [1][2] refer to the conditions in our Solar system due to the activity detected on the Sun´s surface (like solar winds/flares/holes and coronal mass ejections (CMEs)). The effects of these phenomena can be observed in the degradation of spacecraft communication and in general their reliability, possibly damaging power distribution networks, or disrupting radio communication [3][4], but also poses risks to human health in space and on the Earth. Key challenges are: (i) supporting decision-makers with immersive visualization tools to define quick countermeasures to minimize the threats these phenomena pose and (ii) creating awareness in the general public, since having basic knowledge of such phenomena is fundamental to react to these events.

Recent advances in the computing performance of both virtual reality (VR) and augmented reality (AR) hardware, which can now produce very realistic experiences, show that these technologies are finally powerful enough and cheap to be made available to a larger public. Also, AR and VR headsets are already used productively in several professional areas like surgery, design, and training. These technologies can provide a new level of immersion and understanding of complex 3-dimensional datasets/models and environments.

Modern visualization tools are already exploiting VR and AR to enhance data visualization and exploration, but many existing solutions are limited in functionality, immersivity, or usability. There are several challenges to overcome when developing immersive experiences. First, a single tool supporting domain experts and public end-users can be tricky to define since these two user groups have vastly different requirements and goals. Second, even if hardware advances can produce immersive experiences, the increasing data size poses further challenges in processing and visualizing such data in a VR setup. Furthermore, the development of extensible tools that run on commodity hardware and high-end VR setups, such as cave automatic virtual environments (CAVEs) [5] and domes, often forces developers to make compromises when it comes to interactivity. This means that these setups and devices must handle different amounts of data and support several human-computer interfaces. Last, convincing immersive experiences also require novel, intuitive interaction metaphors for data exploration that can leverage the spatial dimension of the data.

In this paper, we present the results of an activity in support of ESA's Space Safety Program (S2P), referred to as CosmoWeather project, in which such challenges have been analysed and led to the definition of a prototypical system for the immersive visualization of space weather phenomena. During this project, the processing pipeline starts directly from the output of a space weather simulation (for example coming from ENLIL [6] or EUHFORIA [7] models), and executes a data preparation step in which the data is reduced to the specific needs of a given visualization tool. Then, such post-processed data is visualized by two distinct visualization tools: one targeting domain experts (referred to in the remaining of this paper as EXPERT or EXP prototype), and the latter targeting the general public (referred to in the remaining of this paper as EDUCATIONAL or EDU prototype).

The EXP prototype is developed on top of CosmoScout VR [8], an open-source visualization tool. CosmoScout VR utilizes optimized rendering algorithms running on a GPU for interactive visualization of large-scale data. Existing features include level-of-detail (LOD) rendering solutions based on a quadtree data structure and the HEALPix projection [9] for the visualization of topographic information with imagery, measurement tools to analyse the surface, and human-machine interfaces to browse and explore the visualization from a simple desktop PC to huge VR systems like domes and CAVEs. Since for this prototype the input data size and density are considered to be large, a key task is to provide an interactive and intuitive way to visualize key information from the simulation output. The tool assists domain experts in choosing relevant data to be analysed and in the end enhance their analysis pipeline.

The EDU prototype targets the dissemination of the project concepts to the general public. Thus, it has to be executable on commodity hardware. We target low-cost, widely-used VR devices such as the Oculus Quest 2, while also providing cross-platform compatibility. We develop this application using the Unity game engine, which allows us to target multiple platforms and deliver the same content on VR head-mounted displays (HMDs), web browsers, or even smartphones. Considering the needs of the target user group, an educational VR experience of SWE phenomena is provided, focusing only on key features of the SWE visualization. Thus, the main purpose of the EDU prototype is to educate the user about the general principles of SWE phenomena.

The paper is organized as follows. In Section 2, we review existing efforts at using VR and AR in the domain of space weather phenomena and the activities of the major space agencies in using immersive environments. In Section 3, we present the user stories implemented in the prototypes. In Section 4, we give an overall system overview, in which the basic system components and the effort in effectively processing and immersively analysing and visualizing space weather phenomena are described. In Section 5, we describe the data preparation step. In Section 6, we describe the software architecture of the EXP prototype and the features that have been implemented to support immersive space weather phenomena visualization. In Section 7, we describe how the EDU prototype has been defined and how the selected user stories have been implemented. Lastly, in Section 8, we give some remarks and directions for future work.

2. State of the Art and Related Projects

In this section, we review first existing related works in the domain of space weather phenomena evaluated using either VR or AR techniques, and then, in section 2.1, we describe key activities of ESA and NASA linked to the use of AR and VR at disseminating space-related events.

Space weather has been widely studied in the literature [1][2][10][11][12] and nowadays there is an increasing interest to understand the implication of space weather conditions in space missions, on space devices, and on everyday life [3][4][13][14]. Immersive visualization (to the extent of VR and AR techniques) has been applied extensively over the years [15][16][17] and in some cases has been used to enhance collaboration between space engineers [18].

Another interesting aspect to consider is how to make a complex topic such as space weather conditions appealing and interesting to a broader audience. In the literature this could be referred to as "storytelling" and the use of VR and AR techniques has been analyzed from both a psychological and technical perspective [19][20][21][22][23]. In [19] it is discussed how haptics combined with visual feedback contribute to the sense of presence in VR. In [20], storytelling has been explicitly paired with VR to create an awareness-raising experience about climate change. It further confirms how VR experiences increase interest, concern, or knowledge about such issues. In [21] it is concluded that immersiveness is crucial for a dramatic storyline narrative and depends on time density. In [22] it is shown that storytelling facilitates knowledge gain in learning environments. In [23], guidance and examples on "research dissemination" and "education" embedded in interactive articles are provided, analogue to our case of combining spoken and written text with interactive content in VR. The related work mentioned above supports the notion that storytelling and VR are suitable methods to convey knowledge for educational purposes in an engaging, meaningful way.

Restricting now the scope to scientific and immersive visualization applied to space weather conditions, we have identified those literature works that are related to the CosmoWeather project.

In [24], a desktop VR software application designed for the stereoscopic display and interactive manipulation of 3D heliospheric volume data, such as solar wind density, velocity and magnetic field has been presented. The Volume Explorer software exploits stereo and perspective views, and animations of time sequences. In [25], a three-dimensional (3D) visualization system with a glassless stereoscope was developed for the real-time numerical simulator of the interplanetary space-magnetosphere-ionosphere coupling system, adopting the 3D magneto-hydrodynamic (MHD) simulation code. In [26], volumetric rendering approaches with improved performances are described. This thesis can be considered as the basis of the OpenSpace[1] visualization tool. Custom processors have been developed for the open-source volumetric rendering engine Voreen framework[2] to load and visualize science models provided by the Community Coordinated Modeling Center (CCMC) at NASA Goddard Space Flight Center (GSFC). This overall provided a stable and well-supported software base on which building the prototype for rendering ENLIL[3] and BATS-R-US[4] models. An extension of this work has also been discussed in [27], where the design and implementation of space weather related phenomena are described.

[1] https://www.openspaceproject.com/
[2] https://www.uni-muenster.de/Voreen/
[3] https://www.swpc.noaa.gov/products/wsa-enlil-solar-wind-prediction
[4] https://ccmc.gsfc.nasa.gov/models/modelinfo.php?model=BATS-R-US

In [28], a system to analyze and contextualize simulations of coronal mass ejections (CMEs) is proposed. The system is a multi-view system providing visualizations to:

1. compare ensemble members against ground truth measurements,
2. inspect time-dependent information derived from optical flow analysis of satellite images, and
3. combine satellite images with a volumetric rendering of the simulations.

VR is never mentioned in the paper, but some techniques presented can be linked to it, and thus we classify this research as a “desktop VR application”. Similarly, to [26], also here the Voreen framework has been used for creating the visualization concepts. In [29], a real-time visualization method for a 3D magnetic field based on AR technology is presented. It enables learners to freely and interactively move the magnets in 3D space and observe the magnetic flux lines in real-time. This prototype relies on Unity engine[5] for the rendering part and on Vuforia AR SDK [6] for enhancing the AR experience. In [30], an approach for the visualisation of space weather magnetohydrodynamic (MHD) model outputs is defined to employ VR. The aim is to be able to visualise the propagation of the plasma from a solar event as well as its interaction with background solar wind in a 3D environment, increasing knowledge about the propagation and unfolding of CMEs that pose a threat to human activities both on Earth and in space. For presenting CMEs generated with the European EUHFORIA MHD model, the Unity game engine has been used.

These related works overall prove that there have been (especially in recent years) efforts at visualizing space weather phenomena, but except for a few cases [27][28], that are defined in well-known open source projects, like OpenSpace, the other studies show prototypical developments that have not been released in the public domain [25][28][29][30] or are in an abandonware state [24]. For the loading and rendering part, state-of-the-art libraries, like Visualization ToolKit (VTK) [31], are generally used [25]. Also, several times [26][27][28][29][30] existing rendering frameworks or engines, like Voreen or Unreal 3D, have been adopted to simplify the prototyping/developing activity. For the volume rendering part, mostly there are two strategies adopted, either rendering the gridded output from the simulation [25][29] or using a ray-marching strategy in the Cartesian world space [28]. In [28], this latter strategy is used since the ENLIL model does not generate a gridded output, and the ray-marching strategy speeded up the visualization pipeline by avoiding the pre-processing gridding step.

Regarding the expected interaction with the data, all systems allow an inspection of the simulation output, while the interaction with the data (i.e., by letting the user “play” with it by thresholding and customizing the quantity visualized) is not supported. Most of these systems are envisioned for desktop use [24][25][26][27][28], and only two are designed and developed for VR and AR environments [29][30]. Also, few methods have been defined for dissemination purposes [26][27][28] or student group studies [29].

2.1 Related Projects

In this section, we present past and current projects dealing with the visualization or dissemination (for creating awareness) of space related events. The two clear main players are the US and EU space agencies. Both have released apps in the public domain for visualizing certain aspects of space related activities, but they are not releasing applications or tools tackling the space weather visualization problem in VR or AR.

2.1.1 NASA Apps for Smartphones, Tablets and Other Digital Media Players

NASA created several apps for disseminating the “universe” awareness. Such apps cover a wide variety of topics, including the International Space Station (ISS) and space shuttles, the Solar system / Universe / Earth, and in general NASA technology. The way they disseminate is through virtual tours, videos, and podcasts. AR has been used for visualizing spacecrafts on Android phones (Spacecraft AR[7]), while space weather conditions are visualized through two apps for iPhone (Nasa Space Weather App[8] and NASA Space Weather Media Viewer[9]). None of these apps employs VR techniques.

[5] https://unity.com/
[6] https://developer.vuforia.com/
[7] https://play.google.com/store/apps/details?id=gov.nasa.jpl.spacecraftAR
[8] http://itunes.apple.com/us/app/nasa-space-weather/id422621403?mt=8
[9] https://apps.apple.com/us/app/nasa-space-weather-media-viewer/id398687618

2.1.2 ESA Activities / Apps Supporting VR and AR

ESA extensively relies on AR/VR for training astronauts for life on the International Space Station, practicing, for example, space walks and operating payloads in zero-gravity, without leaving the Earth ground[10]. Other activities include the exploration of recreational information for tourists (CapeOutdoors3D[11]). This web app overlays Sentinel-2 maps of South Africa with location-tracked information for tourists. Another ESA co-funded project, Wildwego[12], takes a similar approach to enhance the experience of visitors in national parks. It intends to encourage a greater appreciation for nature conservation with an AR-based game that offers information about the flora and fauna surrounding the players as they travel through the park, with the option to share the experience with friends.

3. Implemented User Stories

In this section, we describe the user stories that have been implemented in the two prototypes during the CosmoWeather project.

3.1 EXPERT Prototype User Story: Monitor Coronal Mass Ejection Propagation in Space

Coronal mass ejections (CMEs) are a large release of plasma and associated magnetic energy from the Sun's corona. In a typical CME event, between 109 to 1010 tons of plasma is released into space with velocities between 250 to 1000 km/s. The CME size and frequency vary from several times a day to a few CMEs during the week.
CMEs are known to be the main drivers of space weather phenomena [32]. Earth-directed CMEs can compress the Earth's magnetosphere and reconnect with the Earth's magnetic field. Also, CMEs often produce interplanetary shocks, causing geomagnetic storms.

The most important forecast is the CME arrival time and its geomagnetic effectiveness. Among the most important simulation tools, there are the Magneto Hydro Dynamics (MHD) codes, such as ENLIL and EUHFORIA.

As the main user story for the project, a domain scientist wants to visualize the CME simulated with the EUHFORIA code. The simulation provides information about density, velocity, and magnetic field at different simulation times. An example of the EUHFORIA output is depicted in Figure 1 showing the radial velocity and density on two planes: the left panels depict the solution on the heliographic equatorial plane, while the right panels show the meridional plane that includes Earth. The reference system uses Heliocentric Earth Equatorial (HEEQ) [33].

Scientists then interact through the user interface by selecting a specific scalar value to render. Once rendered, for example, they can then threshold the values by just selecting specific ranges of the scalar.

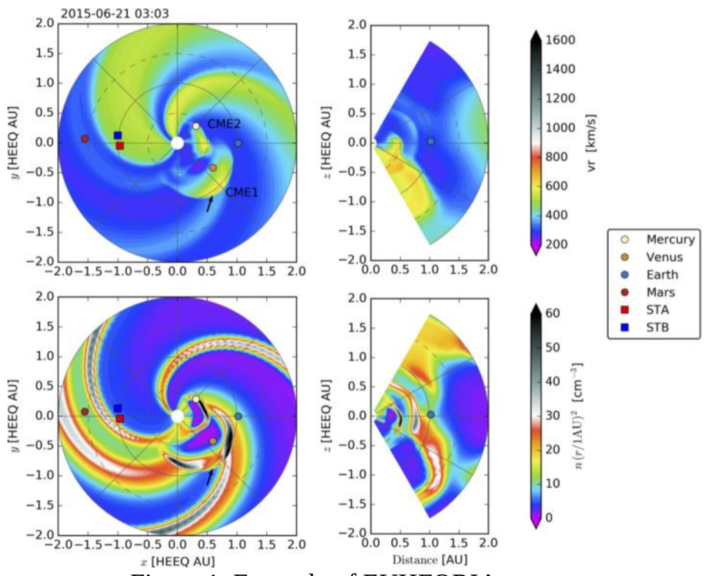


Figure 1. Example of EUHFORIA output.

10 https://www.esa.int/Enabling_Support/Preparing_for_the_Future/Discovery_and_Preparation/Help_us_boost_human-machine_interfaces_to_make_science_fiction_a_reality
11 https://business.esa.int/projects/capeoutdoors3d
12 https://business.esa.int/projects/wildwego

As part of this project, we intend to provide a three-dimensional view of these quantities. Figure 2 shows an example of three-dimensional visualization of the mass density values obtained from EUHFORIA.

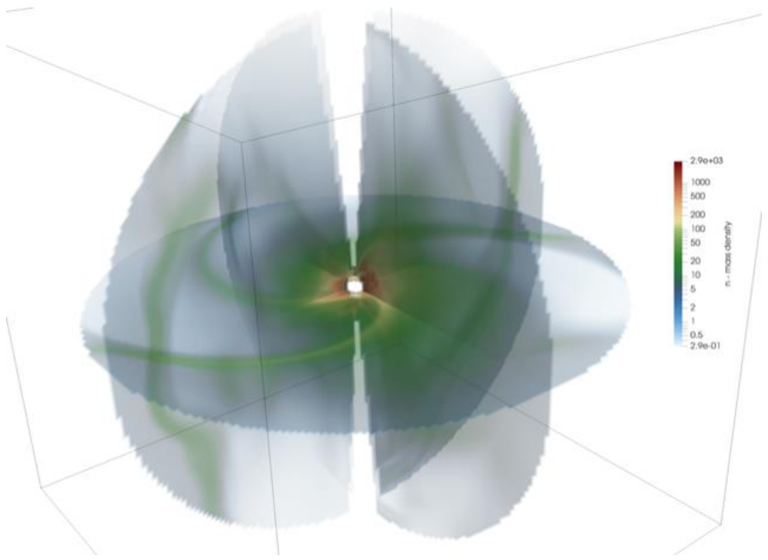


Figure 2. Three-dimensional visualization of the mass density obtained with EUHFORIA.

3.2 EDUCATIONAL Prototype User Stories

The people usually concerned with space weather are domain experts who study space weather phenomena and try to predict their occurrence and the effects they might have on human life and infrastructure on Earth and in space. However, since space weather can affect the lives of everyone on Earth, we need to create awareness for this topic in the general public and educate people about potential threats from space weather events. This requires appropriate means of dissemination to non-experts. This was the motivation behind the development of an educational prototype.

During the project, we defined the following user stories for the EDU prototype.

3.2.1 User Story "Guided Tour"

In the first user story for the EDU prototype, we imagine a non-domain expert, who is interested in space and technology. This user wants to know how and where space weather originates from, how it can influence humans in space and on Earth, and what consequences extreme space weather phenomena can have. We assume the user does not have profound knowledge of physics, and therefore wants to experience these phenomena in a simplified, but visually engaging way, without having to decide what to watch and in which order.

Our proposed approach is a journey through space, starting close to the Sun and following the expansion of space weather phenomena from the Sun until they arrive at Earth. The users' point of view is guided along this path and visual effects with animations and an audio commentary explain the evolution of space weather phenomena along the way.

The last scene of the guided tour also contains an "*exploration mode*", where the user can get additional information on different space weather phenomena. In this scene, referred to as the "*Gallery*", the user has more options and freedom of movement while exploring additional content, like infographics and textual reports related to space weather.

3.2.2 User Story "Executive Summary"

Our second user story for the EDU prototype targets members of the press or decision-makers, who want a short introduction to space weather without having to experience the full "Guided Tour".

It consists only of the most salient points, compressing the whole presentation to a time frame of approximately three and a half minutes (not including additional time for exploration in the Gallery). The story still needs to be consistent, informative and visually engaging, while speeding up the journey between the Sun and the Earth. The last scene of the executive summary is again the *Gallery* where users can spend as much time as they want in the exploration mode.

An important requirement for both user stories is to avoid motion sickness, especially during fast travels between such huge celestial bodies like the Sun and the Earth, which are about 149.60 million kilometers away from each other. We wanted to create a meaningful VR experience, in which we not just look at a small model of the solar system but can observe the Sun and the Earth at a large scale, while at the same time travelling between these two (which would be over a long distance) in a fast and smooth way. To achieve this, the Earth-Sun system is represented as a model where diameters and distances are in relative scale to each other only when the user is near the Sun or the Earth, but not when traveling from Sun to Earth. Instead of moving the point of view for the user over a large distance (which might induce motion sickness), we scale up the model of the Earth, and scale down the Sun, while moving the user only a short distance, thereby avoiding the onset of severe motion sickness.

4. The CosmoWeather System

In this section, we give an overview of the system design and the three components forming the prototype:

- Data preparation
- EXP prototype
- EDU prototype

Even if the system is formed by components that are completely independent of each other (especially the two visualization prototypes), we create a single system design (see Figure 3) since both prototypes rely on and use the files generated in the pre-processing data preparation step.

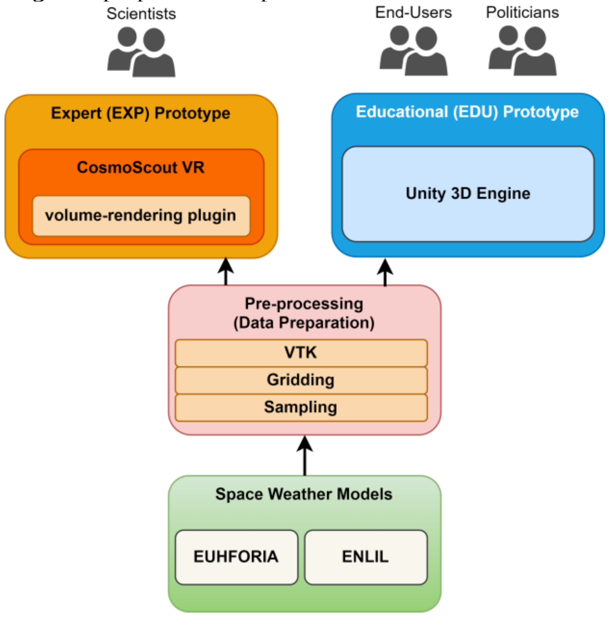


Figure 3. The CosmoWeather system design. From the output of a space weather model, the data preparation formats the data to be rendered in the EXP and EDU prototypes.

As depicted in Figure 3, the pipeline starts from the output of a space weather simulation, e.g. generated by the EUHFORIA or ENLIL models, that targets the evolutions of space weather phenomena, like a coronal mass ejection. Then, the first step performed in the system is a data preparation step in which the data is formatted and prepared to the needs of a visualization prototype. This includes a gridding of the simulation output (if this is not already performed by the specific model) with varying resolutions to have tailored outputs for the EXP and EDU prototypes, and a sampling/reconstruction step in case the data provided by the model just includes a few planes.

As underlying tools of this pre-processing step, we rely on Python scripting, since it provides a relevant number of existing libraries that can be adopted for preparing the data for visualization, and the Visualization ToolKit (VTK) [31], represents the de-facto file standard in the field of scientific visualization. Further details of the data preparation step are given in section 5.

Once the data has been prepared, the two visualization prototypes then load and visualize the data. The EXP prototype is based on CosmoScout VR and its volume-rendering plugin (described in section 6). The target audience of this prototype is composed of domain scientists. Thanks to its plugin-based architecture, it can be used on several VR systems, from a "classical" desktop VR to immersive VR settings, like domes and caves. Then, the expected interaction directly reflects this scalability and goes from a classical mouse+keyboard (of desktop VR environments) to VR positional tracking and glasses.

The goal of the EDU prototype is the dissemination of space weather phenomena to a broader audience (that includes space enthusiasts and laymen). Hence, the prototype is implemented using the Unity Engine, which facilitates the creation of multiple versions for different hardware devices. A description of this prototype is given in section 7.

5. Data Preparation

As mentioned earlier in this section, a pre-processing step is required to process the output data from the space weather models into the CosmoWeather system. As a driver for our architecture design, we focus on the EUHFORIA and ENLIL space weather models. In the case of the EUHFORIA model, the main output we obtain from the code is stored in a Python zip file. This file contains information on different simulation quantities at a given timestep defined on spherical coordinates in a compressed format. The visualization of a complete space weather event requires accessing several Python zip files containing the evolution of the event. The Python zip files include the information on 5,621,760 grid points distributed on a spherical grid with dimensions 512x61x180 for the following quantities:

- BClt = magnetic field component along the latitude direction
- Blon = magnetic field component along the longitude direction
- Br = magnetic field component along the radial direction
- P = pressure (scalar value)
- N = plasma density (scalar value)
- vclt = fluid velocity component along the latitude direction
- vlon = fluid velocity component along the longitude direction
- vr = fluid velocity component along the radial direction

In this case, the total file size (for a time step) is 359,9 MB. For effective usage of several time steps, and to use the model output in the EXP and EDU prototypes, we need to pre-process the EUHFORIA VTK file set. A similar approach can be used to extract the data from ENLIL space weather predictions available on the NOAA space weather prediction center website[13]. In this case, the ENLIL dataset consists of a unique NetCDF file containing information about mass density and plasma speed at different time steps on three different surfaces. To decode the information stored in the ENLIL NetCDF, we use the enlilviz Python module[14].

The different pre-processing steps are shown in Figure 4. In this pipeline, first, a coordinate transformation is performed, then selected properties are extracted, derivative properties are computed, and lastly, a grid resolution is selected considering the requirements of each visualization prototype.

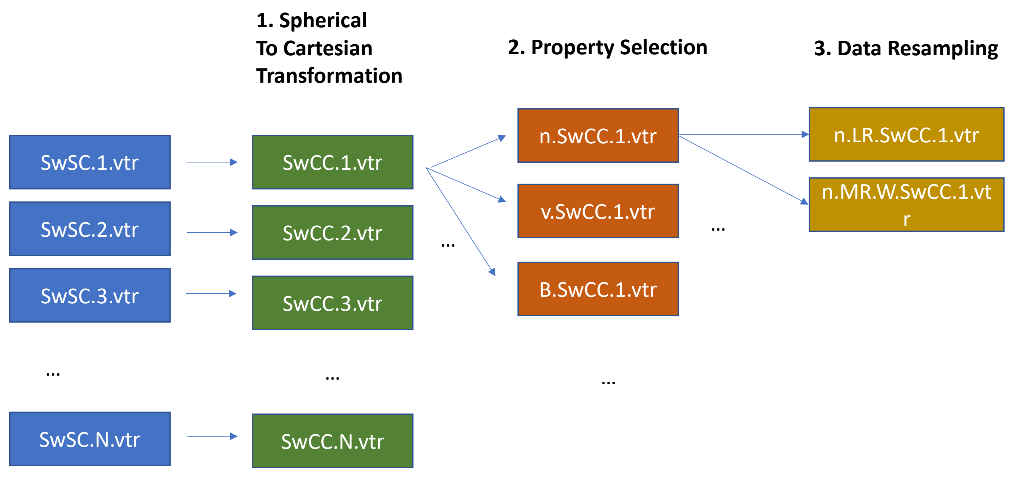


Figure 4. Steps of the data preparation pipeline.

[13] https://www.swpc.noaa.gov/products/wsa-enlil-solar-wind-prediction
[14] https://pypi.org/project/enlilviz/

The first step performed in the pre-processing pipeline is the Spherical to Cartesian coordinates transformation, by using the following equations:

- $x=\rho sin(\varphi)cos(\theta)$
- $y=\rho sin(\varphi)sin(\theta)$
- $z=\rho cos(\varphi)$

where (x,y,z) are the Cartesian coordinates, ρ is the radial distance, φ is the latitude and θ is the longitude. Notice that this transformation is just required in the EDU Prototype, since rendering directly the Spherical coordinates in the Unity engine is not currently supported. Conversely, in the EXP Prototype, based on CosmoScout VR, this transformation is not needed, and the original spherical coordinates are loaded.

For performing this transformation, we developed a Python script, in which we create an additional file in VTK format with the data stored on a Cartesian grid in binary format. We note that this transformation increases the data size of the original file as we now need to include information about the grid coordinates. As dependencies, we rely on the VTK module to import the dataset and the Numpy module for the Cartesian grid transformation. In addition, we designed and implemented the conversion of vector components of the magnetic field and velocity in Spherical coordinates to Cartesian coordinates by using trigonometric projections.

The second step of the pre-processing pipeline is the selection of a property. Each space weather model outputs different quantities (e.g., the density, the fluid velocity, pressure, magnetic field, etc.), and this output is either a single file, encoding all time steps (ENLIL model), or multiple files, each encoding a single time step (EUHFORIA model). In the case of the EUHFORIA model, where each file encodes the full volume, this results in a very large file that is hard to load and handle at run-time in the visualization prototype. The overall rationale for doing this is to have multiple files encoding only the quantities of interest and a single time-step to be loaded by the selected prototype to reduce the I/O times.

The third step is to provide different resolutions for different quantities. Our original data is high resolution, and we derive additional VTK files containing features at decreased resolution. The desired resolution is selected in the Python script and is tailored considering the requirements of a given visualization prototype.

6. Expert (EXP) Prototype

This prototype is based on CosmoScout VR [8] and, in the following, we describe the main software structure, the key plugins, and which features have been developed to support the "monitor coronal mass ejection propagation in space" use case (as described in Section 3.1).

6.1 Software Structure

CosmoScout VR is a software whose development started 10 years ago and evolved from a level-of-detail (LOD) planet-scale renderer to a multi-scale visualization of the entire Solar system and beyond. Figure 5 gives a high-level overview of the involved components: CosmoScout VR is based on external libraries, such as the ViSTA framework[15] [34] for I/O and cluster synchronization, and SPICE[16] for positioning celestial bodies. SPICE was chosen as it is standard in the space industry. ViSTA was not only chosen for historical reasons but also because it is completely open source and thus introduces no license or royalty issues.

[15] https://www.vr.rwth-aachen.de/software/ViSTA/
[16] https://naif.jpl.nasa.gov/naif/toolkit.html

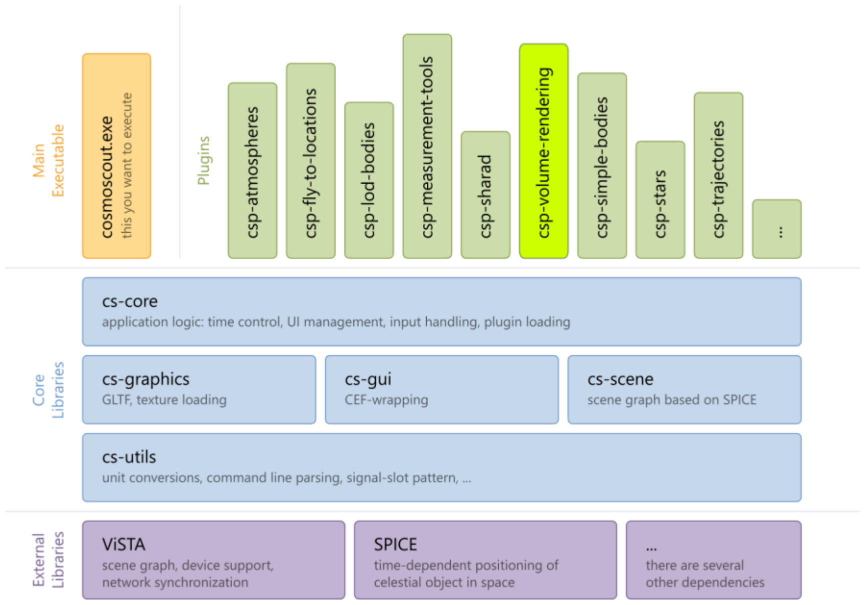


Figure 5. CosmoScout VR current core libraries and dependencies.

The core functionality of CosmoScout VR is split into five core libraries which are built on top of these external dependencies: *cs-utils* contains frequently used functions such as conversions between different units, latitudes, or time formats. The SPICE-wrapping for positioning of celestial objects is implemented in the *cs-scene*. The classes required to build HTML-based user interface elements are implemented in the *cs-gui*. The rendering loop is part of ViSTA, but *cs-graphics* contains specific classes required for rendering 3D objects, shadow computations, and high-dynamic-range (HDR) rendering. Finally, *cs-core* contains high-level functionality using classes of the other libraries, as well as the algorithms for 3D navigation.

The detailed presentation of such core libraries is out of the scope of this document, and thus in the following we briefly describe the core component that is going to be indirectly "affected" by the prototype development, namely, *cs-gui*.

The *cs-gui* core library contains classes required to build a graphical user interface (GUI) based on web technology such as HTML, CSS, and JavaScript using the Chromium Embedded Framework (CEF4)[17]. There are several advantages and disadvantages of using such technology in a virtual environment. The key advantage is the rich ecosystem of open-source libraries for creating all kinds of GUIs allowing developers to easily create complex, interactive GUI elements such as graphs, timelines, or 2D maps. Furthermore, the development cycle of the user interface is comparably fast as changes can be previewed in a web browser without any recompilation of the software. GUI elements can either be placed in screen space or can be attached to scene graph nodes. The former is used when CosmoScout VR is started in desktop mode, the latter is useful in VR where the GUI is positioned in 3D space relative to the user. Each GUI element runs in a separate process, and the CosmoScout VR executable and the plugins communicate with it via inter-process communication. The GUI of CosmoScout VR consists of several core components which can be extended by plugins. An expandable sidebar contains most of the configuration possibilities, and at the bottom left, an integrated JavaScript console allows for interactive scripting. Another central component is an interactive timeline (see Figure 12), which allows the user to adjust the simulation speed, jump to specific time points, or adjust the current simulation time continuously by dragging the timeline.

6.2 The Volume-Rendering Plugin

The aforementioned core libraries do not provide any visualization modules. This is rather implemented as plugins which are loaded by the CosmoScout VR executable at runtime. If CosmoScout VR is started without any plugins, the user only sees a black screen with a minimal user interface. Loading specific visualization functionality from plugins not only reduces development time since it can be reloaded at runtime but also allows for the integration of novel data visualization methods without affecting core components of CosmoScout VR. Plugins can bring in their third-party dependencies, use all functionality of the core libraries, and modify the user interface. By design, plugins cannot communicate with each other. This has not been required yet and simplifies the

[17] https://github.com/chromiumembedded

implementation significantly as plugins do not depend upon each other. Plugins currently supported in CosmoScout VR include basic visualization pipelines, like the rendering of simple bodies, planet trajectories, and stars, as well as more complex texturing or data management, and LOD terrain rendering. Since in this project the development focused on a single plugin, namely *csp-volume-rendering*, we give a detailed overview only of this plugin.

The *volume-rendering* plugin is the main actor in CosmoScout VR when dealing with 3D volumetric data. The main goal of this plugin is the handling of asynchronous volume visualization [35]. Computationally-intensive tasks are started from the main thread of CosmoScout VR as asynchronously executed threads to affect the update rate of CosmoScout VR as little as possible. As noted in Figure 6, the plugin has three main dependencies on other libraries: Intel OSPRay, the Vizualization ToolKit (VTK) [31], and parcoords-es.

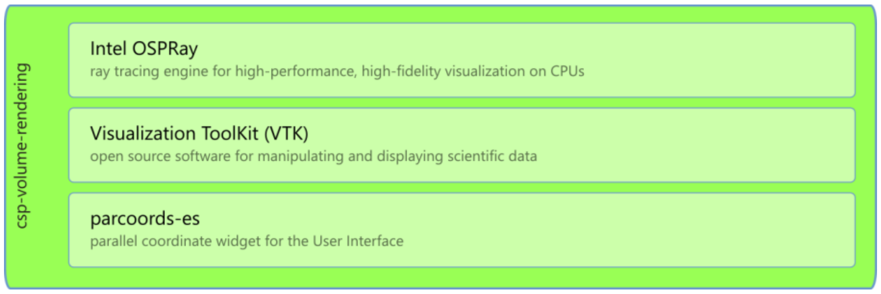

csp-volume-rendering
Intel OSPRay
ray tracing engine for high-performance, high-fidelity visualization on CPUs
Visualization ToolKit (VTK)
open source software for manipulating and displaying scientific data
parcoords-es
parallel coordinate widget for the User Interface

Figure 6. Main dependencies of the *volume-rendering* plugin.

Intel OSPRay[18] (Open source, Scalable, and Portable Raytracing) is a device-independent raytracing API at a high level of abstraction. The standard implementation of this API uses only the CPU for the calculation so it can also be used on computers without a dedicated GPU. This also allows the use of the entire main memory of a computer. The parallelism of the calculation required for efficient ray tracing is achieved through “Single Instruction, Multiple Data” (SIMD) operations of modern CPUs. The *Visualization ToolKit*[19] is used for loading and manipulating the input data, encoded in a VTK-supported format. Lastly, *parcoords-es*[20] is an open-source library in Javascript for rendering a parallel coordinates widget directly into the core component of CosmoScout VR. This latter plots the scalar data encoded in the input data, and also shows for the active scalar a histogram for highlighting the distribution of the values (see Figure 7).

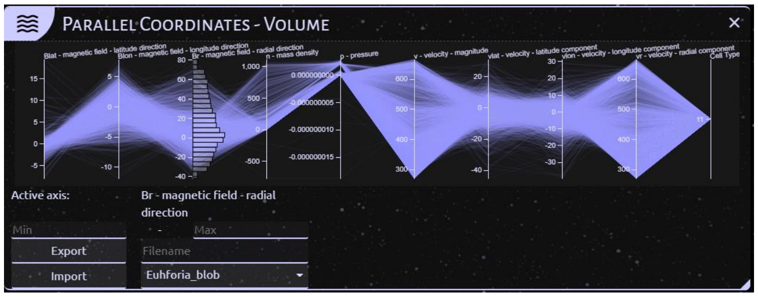

Parallel Coordinates - Volume
Blat - magnetic field - latitude direction
Blon - magnetic field - longitude direction
Br - magnetic field - radial direction
n - mass density
p - pressure
v - velocity - magnitude
vlat - velocity - latitude component
vlon - velocity - longitude component
vr - velocity - radial component
Cell Type
Active axis:
Br - magnetic field - radial direction
Min
Max
Export
Import
Filename
Euhforia_blob

Figure 7. Parallel coordinates plot.

The architecture of the plugin can be logically divided into three main components (Figure 8): (i) the core of the plugin, (ii) the component for performing the volume visualization and (iii) a component that renders the calculated images in CosmoScout VR. The plugin core is mostly delegated to send instructions (triggered from the user interface) to the other two components. The three components behave in an asynchronous way to allow CosmoScout VR to work properly when the data is, for example, loaded or rendered. The asynchronous behaviour also makes the volume-rendering pipeline more flexible since data loading and rendering can be run independently and start as soon as new data is provided. The data loading part supports level-of-detail rendering, and if multiple levels of detail are available, the lowest level of detail is loaded first. Higher LODs are loaded asynchronously when the timestep did not change for a predefined time. The updates of the view are directly linked to the user’s camera so that the volume

[18] https://www.ospray.org/
[19] https://vtk.org/
[20] https://github.com/BigFatDog/parcoords-es

can be examined from different angles. The rate at which the updates happen depends on the complexity of the data, the used hardware, and the rendering parameters.

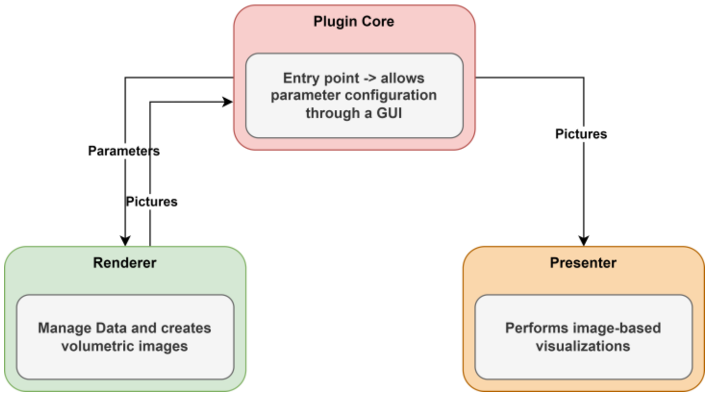

Plugin Core
Entry point -> allows parameter configuration through a GUI
Parameters
Pictures
Pictures
Renderer
Manage Data and creates volumetric images
Presenter
Performs image-based visualizations

Figure 8. Core interaction patterns in the *volume-rendering* plugin.

The user/observer interacts with the volumetric data directly from the plugin GUI. Such GUI is integrated into the sidebar of the main interface of CosmoScout VR and lets the user edit the transfer function, edit the scalar data visualized, tweak rendering options, etc. Figure 9 shows this GUI.

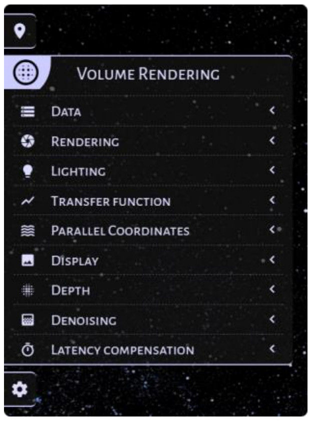


Figure 9. GUI of the volume-rendering plugin.

6.3 Implemented Features

In this project, we extended the csp-volume-rendering plugin[21] in CosmoScout VR to support space weather phenomena and to enhance the interaction a user can perform to analyse and extract features from such data. To tackle this, we performed the following tasks:

- generation of acceleration structures,
- GUI extensions, and
- extension of the volume rendering plugin.

6.3.1 Generation of Acceleration Structures

The generation of acceleration structures is a pre-processing step to be performed one time to create level-of-detail structures that improves the rendering performance in CosmoScout VR. This is required to generate fluent animations and to improve the rendering in HMDs. To generate such structures, we develop a Python script that processes the output data provided by the data processing step (i.e., VTK files). This LOD is encoded using an octree pyramid structure, in which, every level only encodes one eight of the higher-resolution level. Such structures then enable the efficient loading of the data in the prototype, since we load first the coarsest level, and only when the view is stable, the higher resolution levels are loaded. This strategy provides more details when needed, but at the same time it does not affect the fluency of the immersive exploration.

[21] https://github.com/cosmoscout/csp-volume-rendering

6.3.2 GUI Extensions

The volume-rendering plugin provides a Javascript-based GUI, currently including a transfer-function editor, an animation button for sequentially loading all time steps of the simulation, and several rendering and lighting options. Since the density of the data to be rendered is allegedly large, we enhanced the transfer function editor by adding a log-scaling option on the x-axis (see Figure 10). This enhances and eases the understanding of scalar value ranges that are extremely dense. For example, Figure 11 shows the spiral structure of the heliospheric current sheet, highlighted through the transfer function editor, commonly referred to as the *ballerina skirt shape* as it resembles a skirt shape.

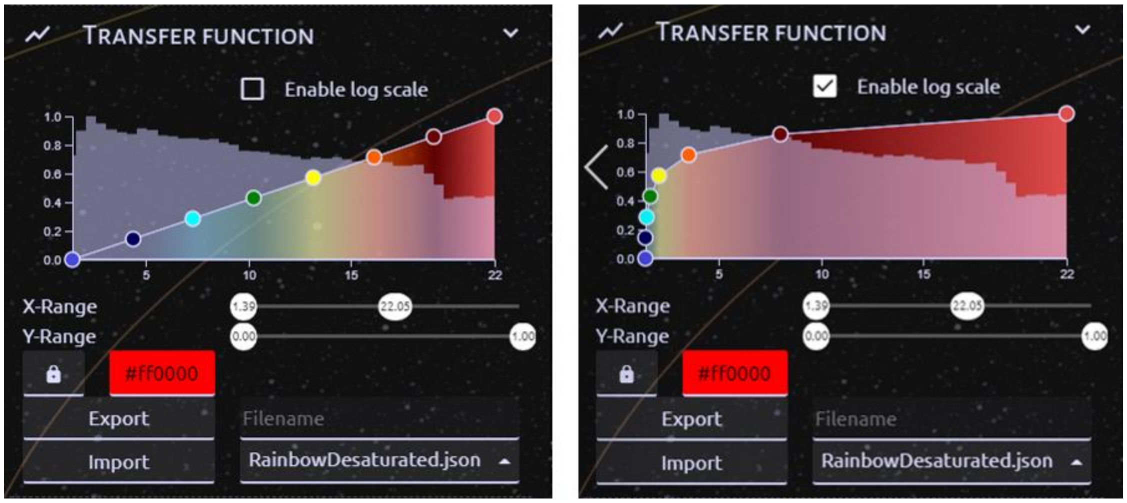


Figure 10. Transfer function editor with log-option disabled (left) and enabled (right).

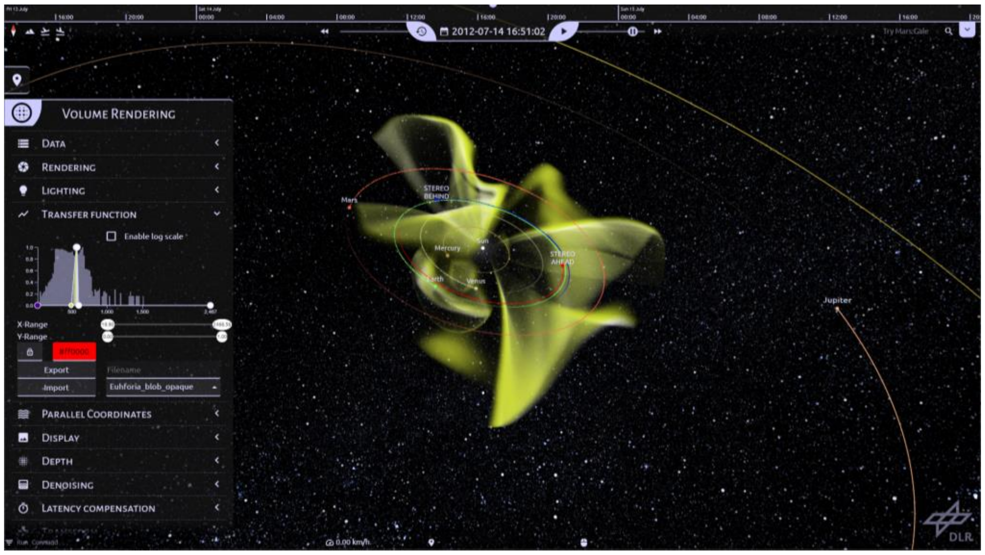


Figure 11. Example of volumetric visualization of EUHFORIA dataset highlighting ballerina outskirt shape.

One last feature to enhance the way the user interacts with the space weather data is to link the already implemented time bar of the main interface of CosmoScout VR (depicted in Figure 12) with the loading of subsequent files (that represents consecutive instants in time). The time bar already provides easy-to-use play, fast-forward and rewind buttons, that let the user easily interact and manipulate the time also with the controllers provided with HMDs in a virtual environment.

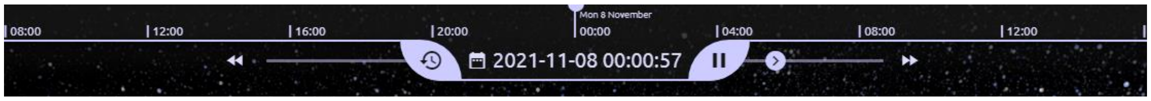


Figure 12. CosmoScout VR's user interface for time navigation.

Previously, the loading of consecutive time steps was provided by a simple play button in the sidebar GUI of the plugin, with which the user could go back and forth between time steps using a simple slider. The main limitation of this approach was that a user could not monitor the time in which a certain space weather phenomenon, like a ballerina outskirt shape, starts or ends. Since the timing information is already encoded in the file name, we developed a procedure that first interprets the date and time encoded as UNIX time-stamp[22], and then loads the corresponding file. This means that a user can now go back and forth in time and immediately realize when a certain phenomenon in a simulation started/ended or reached its maximum peak.

6.3.3 Volume Rendering Enhancements

In this section, we present how we extended the volume-rendering plugin by providing an immersive walkthrough of the data volume.

Previously, the volume rendering plugin supported the rendering of time-varying volumetric data when rendering the whole volume on screen. The volume was always rendered to fit a square at a fixed resolution by dynamically changing the field of view (FOV) to match the edges on the boundary of the volume. This led to a more stable framerate (i.e., the frequency at which consecutive images, or frames, are displayed). However, when the viewpoint is very close to the volume, only a limited portion of this image (with a low pixel count) is visible. Also, the dynamic FOV cannot be computed for viewpoints inside the data volume, since the volume data surrounds the viewpoint and, thus, has no visible edges, resulting in empty images.

Since one of the objectives of this activity is to create an immersive experience for domain experts, we have enhanced the rendering of the interior of the volume data by supporting selective-view-dependent rendering (see Figure 13). The prototype now relies on a fixed square FOV of 120° filling the HMD screen when close to or inside the volume data, and this enables the immersive walkthrough experience. To reduce as much as possible the visualization artefacts (like edges on the image borders) while being in an immersive VR setup, this fixed FOV is configured to be slightly larger than the HMD FOV. Lastly, to enhance the exploration of the inner structures of a CME, the head tracking support is now implemented in the prototype.

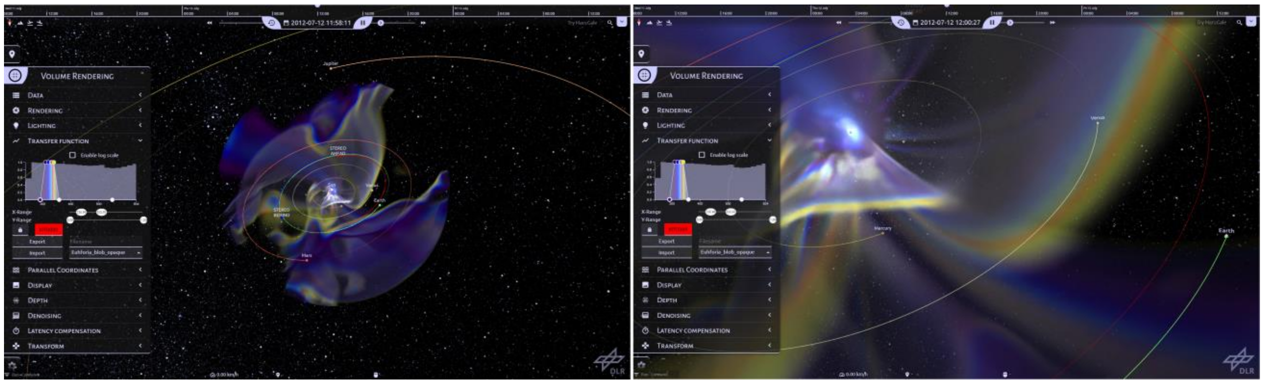


Figure 13. A user can dive into a space weather phenomenon to gain further knowledge.

7. Educational (EDU) Prototype

The educational prototype is realized as a Unity app, building on Unity's editor and associated services. The initial prototype has been developed for the Oculus Quest 2. However, by using Unity's OpenXR plugin, we simplified cross-platform compatibility and potentially support a wide range of platforms (e.g. Windows Mixed Reality, HoloLens 2, Oculus PC + Link, Oculus Quest, SteamVR), which will ease the porting of the app to other devices in the future. A build (executable) for Windows supports the use of multiple different XR devices and has already been tested with Oculus Quest 2 and HTC Vive.

The storyline of the guided tour is split into different *DirectorScenes*, which are saved as *Scriptable Objects*. *Scriptable Objects* can be used as containers to save data, in our case for example the player positions in each

[22] https://en.wikipedia.org/wiki/Unix_time

DirectorScenes, or animation parameters for visual effects. *DirectorScenes* are our concept to structure a cinematic VR experience. A *DirectorScene* holds all components, such as player position, animation parameters, 3D objects, or narration, for a short part of the story (similar to a movie scene). 3D models (of e.g. planets, planes, satellites) in the application are represented as Unity *Game Objects* (the fundamental objects in Unity that act as containers for components and typically represent visual 3D objects) and placed in each *DirectorScene*. The game logic, user interaction, and data management inside the app are implemented using C# as scripting API.

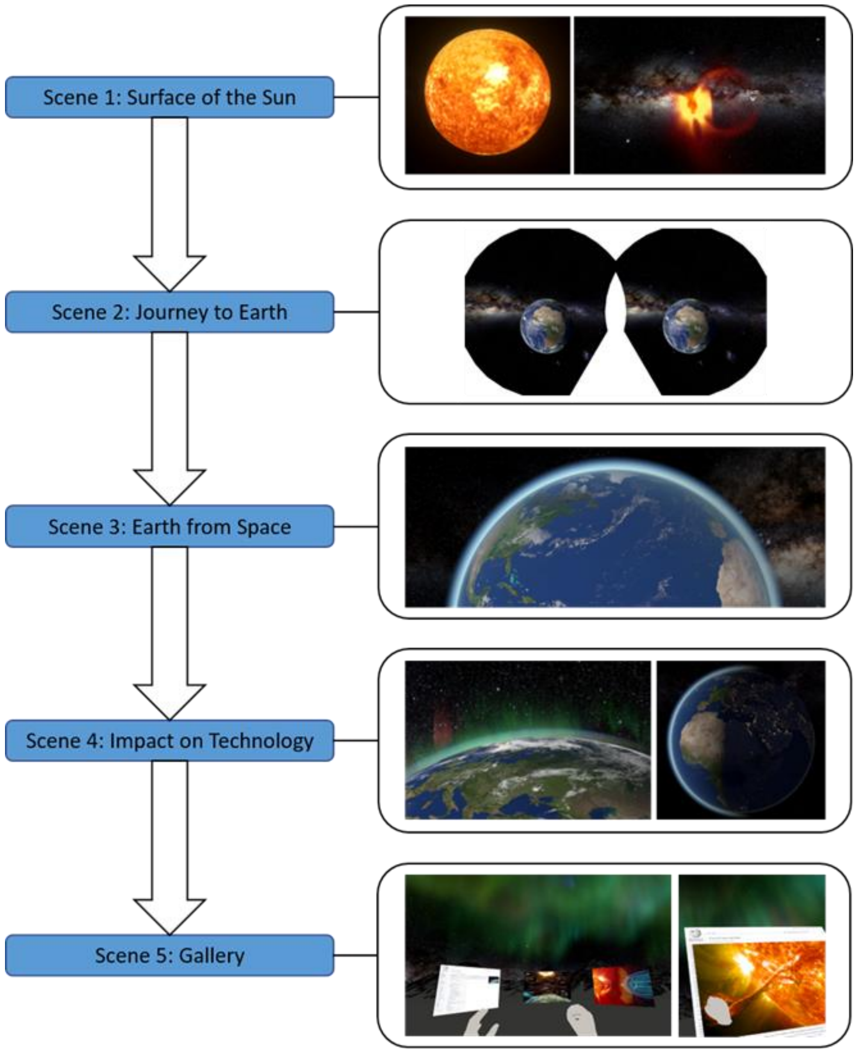
A dedicated *Data Loader* class handles all data import.

7.1 Data Loading

The EDU prototype supports EUHFORIA data as the data source for the CME effect. The data can be imported into the application as 3D volume texture, using a combination of Python (version 3.10) and Unity scripts. The application takes VTR files prepared through our pre-processing step as input. Such a VTR file contains the data of a CME in cartesian coordinates. One or multiple timeframes can be loaded into the EDU prototype to render different steps of a CME. (We recommend files with a resolution of 256x256x256 or 128x128x128 to ensure a smooth experience.) We created a Unity editor script that launches a file browser and allows the user to select a VTR file for import. Upon selection of the VTR file, the application automatically runs a Python script that loads the VTR file and retrieves metadata, such as the file size. This information is passed onto the C# data loader script which allocates the necessary space for the 3D volume texture. The Python script then exports a pre-defined channel from the VTR file to memory, accessible by the C# script. (For the CME effects we use Channel 0, which provides data values representing the magnetic field magnitude.) Finally, the data is written into the 3D volume texture via C# script and can be added to the CME animation script in a *DirectorScene*. Multiple timeframes can be imported as 3D textures this way and added to the visual effect.

7.2 Scenes

The general concept of the application consists of five "*DirectorScenes"* (with their respective game objects and C# scripts), which are played in sequence, as displayed in Figure 14.


Scene 1: Surface of the Sun
Scene 2: Journey to Earth
Scene 3: Earth from Space
Scene 4: Impact on Technology
Scene 5: Gallery
Figure 14. Core scenes of the EDU prototype.

<u>Scene 1: Surface of The Sun</u>

The first scene aims to convey the main principles of the Sun's surface and corresponding phenomena. At first, a close-up of the Sun is shown with the Sun’s surface visualized with an animated texture. When the user moves away from the Sun’s surface, a visual effect shows a CME, based on imported data (Figure 17 (right)) as described above. After the data loading process, each imported timeframe of the CME is available as a 3D volume texture inside the application. Ray marching is implemented in a shader to step through the volume, sample the data and combine it via additive blending with the rest of the scene. A blackbody function is used to map the accumulated samples to a high dynamic range color. (Before displaying the image, Unity’s tone mapping function is transforming the colors into the appropriate color space that can be displayed on the device.) To animate the CME visualization, we modify the opacity of the sampled data. A bias is subtracted from each sampled value. By animating the value of this bias on a curve over time, the CME effect becomes more prominent or vanishes, depending on the accumulated sample values at each position. If multiple timeframes are added to the animation script, they are rendered in sequence.

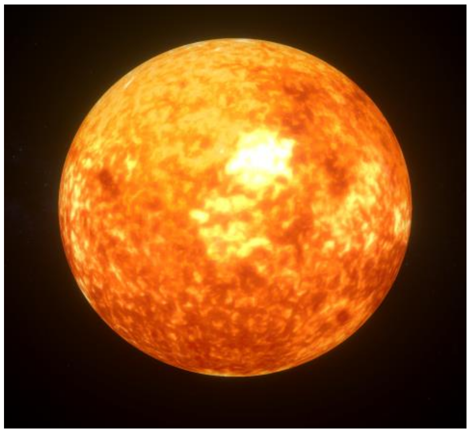
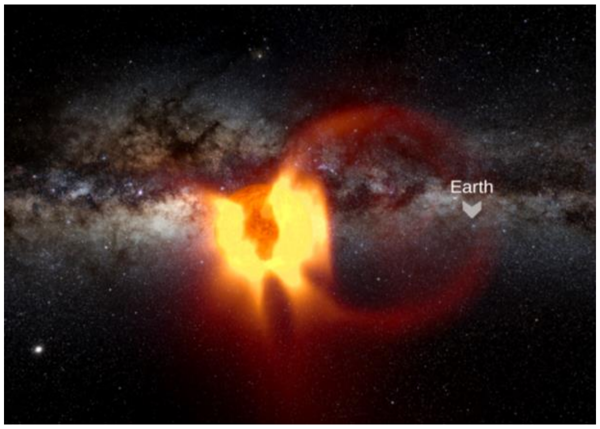



Figure 15. Sun rendering (left) and CME visualization (right) in the EDU prototype.

<u>Scene 2: Journey to Earth</u>

The second scene acts as a transition to Earth (Figure 16).

The travel from Sun to Earth is realized as a movement along a pre-defined path, combined with scaling the Earth up (from small to big) and scaling the Sun down (from big to small). This way, an overly large virtual environment (which would create performance issues) can be avoided and moving the user only over a short distance can also help to avoid severe motion sickness.

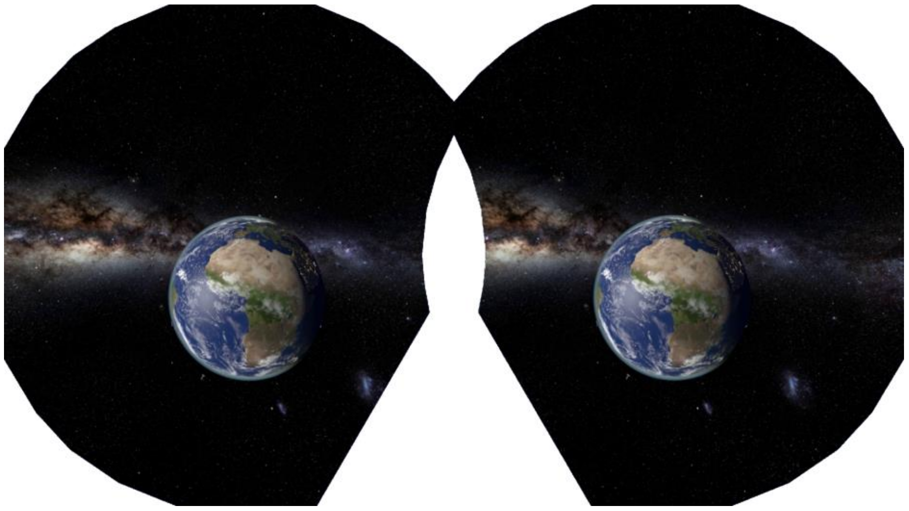


Figure 16. Travel towards Earth (view from the head-mounted display).

<u>Scene 3: Earth from Space</u>

The third scene shows Earth from space, rendered with real planetary maps in equirectangular projection[23]. The Earth's atmosphere is realized with an atmospheric scattering shader[24] that was extended to support rendering in VR applications. Solar flares hitting the Earth are visualized as aurora borealis over the northern hemisphere (the visual effect is based on the Northern Lights Pack by Ian Scilipoti[25]) (see Figure 17).

[23] Solar System Scope, „Solar Textures“ https://www.solarsystemscope.com/textures/
[24] Luke Gamage, "Atmospheric Scattering" https://gist.github.com/T5impact/9edc753a048ea90c7a2bdbaa5c1255d8
[25] https://assetstore.unity.com/packages/vfx/particles/environment/northern-lights-pack-86980

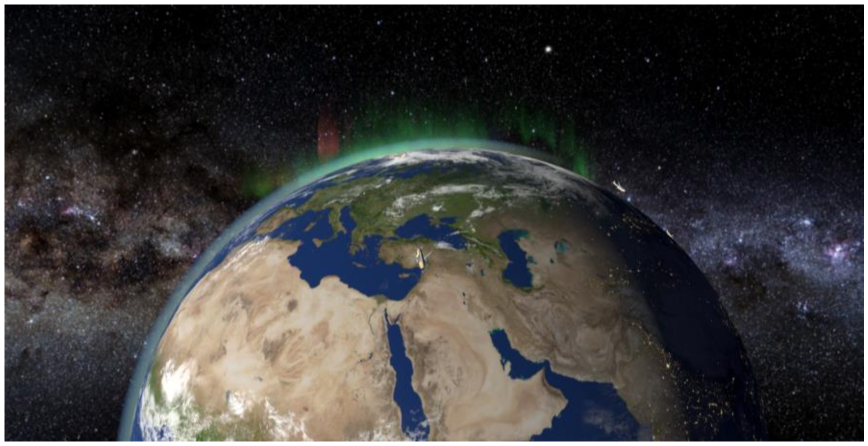


Figure 17. Earth from space with Aurora Borealis effect.

Scene 4: Impact on Technology

The fourth scene is targeted at explaining the impact of space weather on technology on Earth. To that end, the Earth is populated with multiple (freely available) 3D assets, such as satellites, and airplanes.

The impact of space weather on technology is visualized using the two effects. First, we show electric sparks on satellites and planes (caused by a CME hitting Earth) based on an effect from the Unity asset store[26]. Then, the flickering lights on the night side of the Earth are generated with an animated texture[27] (see Figure 18).

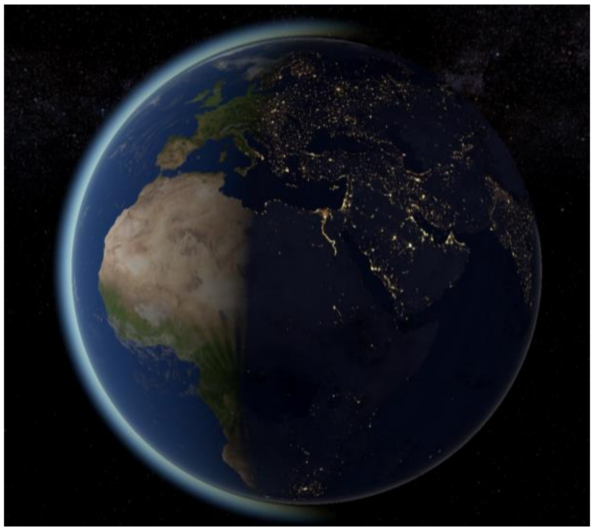


Figure 18. The night side of Earth from space, before going dark.

Scene 5: Gallery

The last scene places the user onto the surface of the Earth, in a night scene with snowy mountains and visualizations of auroras in the sky above. After a few seconds, virtual screens appear around the user and display additional information on space weather events as 2D media (see Figure 19). Such information could, for example, include astronomer sketches, videos of events, eyewitness reports, quotes, 2D images, or scientific articles. For now, we added textual information on space weather events, images and infographics to the Gallery. The virtual screens, displaying this information, are realized as Unity game objects with image planes and text fields. Users can touch a virtual screen and press the trigger on the controller to get further information (e.g. an image to a text, an infographic, or additional textual information) on the respective topic displayed in front of them.

[26] Digital Ruby (Jeff Johnson), "Lightning Bolt Effect for Unity" https://assetstore.unity.com/packages/tools/particles-effects/lightning-bolt-effect-for-unity-59471

[27] Solar System Scope, „Solar Textures“ https://www.solarsystemscope.com/textures/

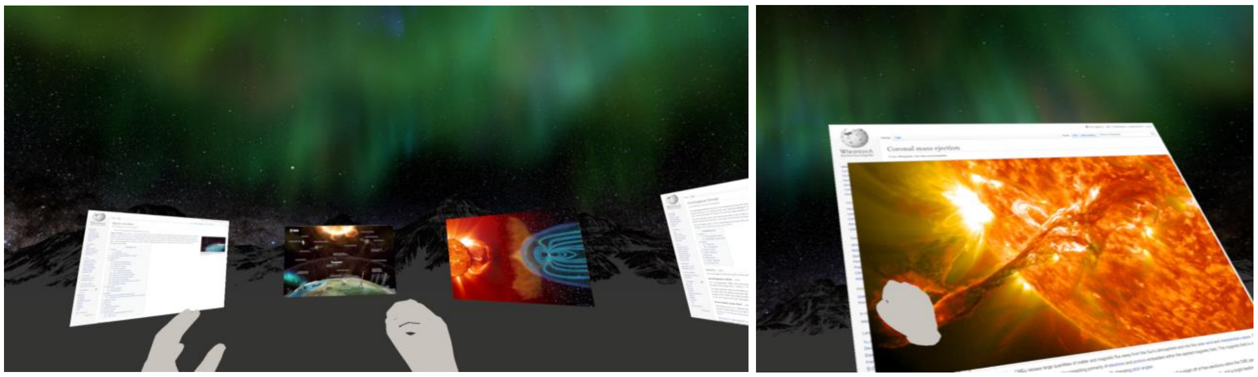


Figure 19. Gallery scene in which the user can get additional information about space weather phenomena.

8. Concluding Remarks

During the CosmoWeather project, the consortium performed a dense prototypical activity to investigate the interactive and immersive visualization of space weather phenomena in support of ESA's Space Safety Program (S2P). The two main objectives of this activity were to:

- provide a prototype of a visualization tool for the immersive visualization of such phenomena to domain experts, supporting high-end HMDs for immersive interaction with the data;
- create awareness in the general public by providing a prototype for an immersive guided visualization of space weather phenomena as an educational experience.

In this prototypical activity, the two prototypes also had to deal with large-scale data provided by the major space agencies through two main models, EUHFORIA and ENLIL. Since the data generated by the two models is fairly different (even if the quantities/properties encoded are pretty similar), there has been a need to effectively pre-process the data generated by the two models such that it is cross-compatible between the two prototypes. Then, the pre-processed data is loaded into the two VR prototypes that are complementary and have different objectives.

The purpose of the EXP prototype is to support domain experts in analysing time-varying data in an intuitive way, supporting diverse hardware, ranging from a classical desktop workstation to an innovative immersive visualization through a head-mounted display (HMD). To facilitate space weather data exploration in VR, the main objectives were to improve the interaction with such data in VR, to ease the navigation through time and to provide an immersive navigation inside the data volume. As a base tool for this development CosmoScout VR has been used, a visualization tool developed by the German Aerospace Center.

The main goals of the EDU prototype are to give to non-domain experts an introduction to space weather, increase interest in this topic, and raise awareness of the possible consequences of extreme space weather events in the general public. As a result of this activity, we developed a prototype implementing two user stories: (i) a "Guided Tour" for a 7-10min experience, which can be developed further by including additional visual effects, and (ii) a shorter 3-4 minutes "Executive Summary", focusing on the main message and visualizing a subset of possible effects of the Guided Tour experience. This prototype was implemented using Unity to develop a flexible framework and a customizable pipeline in which effects and narration for further phenomena can be added as needed.

In the current activity, we focused on the visualization of coronal mass ejections (CMEs) generating from the Sun and heading to the Earth. Since this major phenomenon can cause other phenomena around Earth, it could be interesting to monitor the magnetosphere dynamics as well as Aurora forecasts in the future. Also, other phenomena that could be monitored in follow-up activities are space debris with (if available in the underlying models) trajectory visualization and course changes caused by space weather phenomena (like CMEs). This could create a full-fledged tool ecosystem for the analysis and monitoring of these events.

Concerning the data preparation step, we focused on designing and implementing Python adaptors to convert EUHFORIA and ENLIL output to VTK files that can be visualized in the CosmoWeather prototypes. The pre-processing step required the conversion from Spherical to Cartesian coordinates (and associated interpolation) for the EDU prototype, resizing the dataset (to decrease the size) and writing to VTK files. Different Python modules simplified the development of this pre-processing step. The data preparation is performed offline, given the datasets from EUHFORIA and ENLIL. An increased integration of EUHFORIA (or ENLIL) with the CosmoWeather data preparation step or the establishment of a coupled workflow is the natural next step of this work.

In the current version of the EXP prototype, a user can visualize complex CME simulations in an immersive way in hardware ranging from a classical desktop to professional HMDs, like the Vive Pro. While currently, the interaction with the underlying simulation is fluent and it can be performed intuitively, and data selection and highlighting are currently supported, further enhancements can be envisioned, such as the visualization of iso-surfaces and path lines. Other interesting developments that could be planned in the future are: (i) make the tool compatible with vector data; (ii) enable a user to switch between datasets at run-time; and (iii) enable a communication strategy with the underlying data preparation step, so that the user can compute the properties to process and visualize at run-time.

The current version of the EDU prototype consists of a flexible software framework that can be used to create different storylines for different versions of an immersive guided tour for explaining space weather phenomena, with an exploration mode in the final scene (the Gallery) to provide additional information to the interested user. The current software prototype can be used to create a VR application which can be experienced with either an Oculus Quest 2, an HTC Vive or a Vive Pro. Future work could include the definition of more visual effects based on real data to visualize additional space weather phenomena (e.g. solar flares, sunspots, the Earth's magnetosphere, or magnetic reconnection), and the optimization of the rendering methods for such effects to support low-end stand-alone HMDs. Also, it could be interesting to investigate the development of novel interactive methods for immersive data exploration in AR or mixed reality (MR) settings, including physical objects or hand tracking to explore space weather data. Lastly, we plan also to investigate the use of gamification elements to increase user engagement, enhance the learning experience, and increase awareness of space weather phenomena and their effects on life on Earth in a broader target audience.

Acknowledgements

This work has been supported in the context of a Technology Development Element activity, funded by the European Space Agency (ESA) and the European Space Operations Centre (ESOC), entitled "Augmented/Virtual Reality for Space Safety Use Cases" under contract number 4000136331/21/D/SR. VRVis is funded by BMK, BMDW, Styria, SFG, Tyrol and Vienna Business Agency in the scope of COMET - Competence Centers for Excellent Technologies (879730) which is managed by FFG.

References

[1] Echer, E., Gonzalez, W. D., Guarnieri, F. L., Dal Lago, A., & Vieira, L. E. A. (2005). Introduction to space weather. Advances in Space Research, 35(5), 855-865.
[2] Baker, D. N. (2000). Effects of the Sun on the Earth's environment. Journal of Atmospheric and Solar-Terrestrial Physics, 62(17-18), 1669-1681.
[3] Ferguson, D. C., Worden, S. P., & Hastings, D. E. (2015). The space weather threat to situational awareness, communications, and positioning systems. IEEE Transactions on Plasma Science, 43(9), 3086-3098.
[4] Eastwood, J. P., Biffis, E., Hapgood, M. A., Green, L., Bisi, M. M., Bentley, R. D., ... & Burnett, C. (2017). The economic impact of space weather: Where do we stand?. Risk Analysis, 37(2), 206-218.
[5] Cruz-Neira, C., Sandin, D. J., DeFanti, T. A., Kenyon, R. V., & Hart, J. C. (1992). The CAVE: audio visual experience automatic virtual environment. Communications of the ACM, 35(6), 64-73.
[6] Odstrcil, D. (2004). Enlil: A numerical code for solar wind disturbances. Community Coordinated Modeling Center (CCMC): Hosted Models at a Glance, ENLIL with Cone Model.
[7] Pomoell, J., & Poedts, S. (2018). EUHFORIA: European heliospheric forecasting information asset. Journal of Space Weather and Space Climate, 8, A35.
[8] Schneegans, S., Zeumer, M., Gilg, J., and Gerndt, A., CosmoScout VR: A Modular 3D Solar System Based on SPICE. In 2022 IEEE Aerospace Conference, 2022, (pp. 1-13), IEEE.
[9] Gorski, K. M., Hivon, E., Banday, A. J., Wandelt, B. D., Hansen, F. K., Reinecke, M., & Bartelmann, M. (2005). HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. The Astrophysical Journal, 622(2), 759.
[10] Kivelson, M. G., Kivelson, M. G., & Russell, C. T. (Eds.). (1995). Introduction to space physics. Cambridge university press.
[11] Song, P., Singer, H. J., & Siscoe, G. L. (2001). Space weather. Washington DC American Geophysical Union Geophysical Monograph Series, 125.
[12] McIntosh, S. W. (2019). Space weather: Big & small–A continuous risk. Journal of Space Safety Engineering, 6(1), 48-52.
[13] Cliver, E. W. (2006). The 1859 space weather event: Then and now. Advances in Space Research, 38(2), 119-129.
[14] Green, J. L., Boardsen, S., Odenwald, S., Humble, J., & Pazamickas, K. A. (2006). Eyewitness reports of the great auroral storm of 1859. Advances in Space Research, 38(2), 145-154.

[15] Donalek, C., Djorgovski, S. G., Cioc, A., Wang, A., Zhang, J., Lawler, E., ... & Longo, G. (2014, October). Immersive and collaborative data visualization using virtual reality platforms. In 2014 IEEE International Conference on Big Data (Big Data) (pp. 609-614). IEEE.

[16] Jasmine Y. Shih, Kalina Borkiewicz, AJ Christensen, and Donna Cox. 2019. Interactive cinematic scientific visualization in unity. In ACM SIGGRAPH 2019 Posters (SIGGRAPH '19). Association for Computing Machinery, New York, NY, USA, Article 69, 1–2.

[17] Sicat, R., Li, J., Choi, J., Cordeil, M., Jeong, W. K., Bach, B., & Pfister, H. (2018). Dxr: A toolkit for building immersive data visualizations. IEEE transactions on visualization and computer graphics, 25(1), 715-725.

[18] García, A. S., Roberts, D. J., Fernando, T., Bar, C., Wolff, R., Dodiya, J., ... & Gerndt, A. (2015, March). A collaborative workspace architecture for strengthening collaboration among space scientists. In 2015 IEEE Aerospace Conference (pp. 1-12). IEEE.

[19] Gibbs, J. K., Gillies, M., & Pan, X. (2022). A comparison of the effects of haptic and visual feedback on presence in virtual reality. International Journal of Human-Computer Studies, 157, 102717.

[20] Markowitz, D. M., & Bailenson, J. N. (2021). Virtual reality and the psychology of climate change. Current Opinion in Psychology.

[21] Nakevska, M., van der Sanden, A., Funk, M., Hu, J., & Rauterberg, M. (2017). Interactive storytelling in a mixed reality environment: the effects of interactivity on user experiences. Entertainment computing, 21, 97-104.

[22] Gil, M., & Sylla, C. (2022). A close look into the storytelling process: The procedural nature of interactive digital narratives as learning opportunity. Entertainment Computing, 41, 100466.

[23] Hohman, F., Conlen, M., Heer, J., & Chau, D. H. P. (2020). Communicating with interactive articles. Distill, 5(9), e28.

[24] Wang, X., Hick, P. P., Jackson, B. V., & Bailey, M. (2004, February). Visualization of remotely sensed heliospheric plasmas for space weather applications. In Telescopes and Instrumentation for Solar Astrophysics (Vol. 5171, pp. 280-286). International Society for Optics and Photonics.

[25] Den, M., Kuwabara, T., Ogawa, T., Tanaka, T., Goncharnko, I., & Amo, H. (2006). A 3D Visualization System for Real-Time Space Weather Simulator with a Glass-less Stereoscope. Journal of the National Institute of Information and Communications Technology Vol, 53(1).

[26] Törnros, M. (2013). Interactive visualization of space weather data. In Thesis Reports series (number LiU-ITN-TEK-A-13/015—SE), Linköpings University.

[27] Carlbaum, O., & Novén, M. (2017). Real-Time Magnetohydrodynamic Space Weather Visualization.

[28] Bock, A., Pembroke, A., Mays, M. L., Rastaetter, L., Ropinski, T., & Ynnerman, A. (2015, October). Visual verification of space weather ensemble simulations. In 2015 IEEE Scientific Visualization Conference (SciVis) (pp. 17-24). IEEE.

[29] Liu, X., Liu, Y., & Wang, Y. (2019, March). Real time 3D magnetic field visualization based on augmented reality. In 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR) (pp. 1052-1053). IEEE.

[30] Ntagiou, E.V., Klug, J., Luntama, JP., Sarkarati, M. (2022). Virtual Reality in Support of Space Weather Forecasting. In: Cruzen, C., Schmidhuber, M., Lee, Y.H. (eds) Space Operations. Springer Aerospace Technology.

[31] Schroeder, W.; Martin, K.; Lorensen, B. (2006), The Visualization Toolkit (4th ed.), Kitware, ISBN 978-1-930934-19-1

[32] Srivastava, N., Mierla, M., & Zhang, J. (2021). Space Weather Prediction: Challenges and Prospects. Frontiers in Astronomy and Space Sciences, 230.

[33] Hapgood, M. A. (1992). Space physics coordinate transformations: A user guide. Planetary and Space Science, 40(5), 711-717.

[34] van Reimersdahl, T., Kuhlen, T., Gerndt, A., Henrichs, J., & Bischof, C. (2000). ViSTA: a multimodal, platform-independent VR-Toolkit based on WTK, VTK, and MPI. In Proceedings of the 4th International Immersive Projection Technology Workshop (IPT) (Vol. 14, p. 22).

[35] Fritsch, J., Flatken, M., Schneegans, S., Gerndt, A., Plesa, A. C., & Hüttig, C. (2022). RayPC: Interactive Ray Tracing Meets Parallel Coordinates. IEEE VIS 2021 as part of the SciVis contest.