University of Cape Town

Department of Computer Science

CSC5000W - Masters in Computer Science

2024

VRDAVis: Remote Visualisation of Astronomy Data with a Standalone Virtual Reality Device

Author:
Michaela van Zyl
Student Number:
VZYMIC015

Supervisors:
Prof. Robert Simmonds
Dr. Angus Comrie

Abstract

This dissertation explores the challenges of visualising vast astronomy data cubes using a virtual reality environment, addressing the ever-increasing volume of data collected by radio astronomy instruments. As the amount of data grows year after year—ranging from terabytes to petabytes—radio astronomy researchers face significant difficulties in processing, storing, and analysing this immense data. The visual analysis of the collected data is a crucial part of radio astronomy research. Traditional visualisation tools are often inadequate due to the size and complexity of the data. The Data far exceeds the computational capabilities of devices like laptops or home desktop computers. Researchers are often required to either access specialised systems or analyse small portions of the data at a time. Specialised systems are typically locked to specific locations and inaccessible to many, and segmenting the data can obscure the broader context of a dataset. These points highlight the need for a new approach to overcome the presented limitations.

The objective of this research is to develop a prototype system that enables the visualization of large astronomy data cubes in a virtual reality environment using a standalone VR (Virtual Reality) headset. The system is specifically designed to operate on devices with limited computational power, such as laptops and VR headsets, making it accessible to a wider range of users. The research addresses key questions, including the feasibility of remote implementation, the scalability of the system for handling large datasets, and a performance comparison with existing astronomy visualisation systems.

The VRDAVis system design follows a client-server architecture, where the client-side communicates with the server to visualise large astronomy data cubes. These client devices are either a computer or stand-alone VR headset. The system pre-processes the data into multiple resolution levels, these levels are referred to as mipmaps, to reduce the computational load on the client. The front-end, built as a web-based application, allows users to select data cubes and progressively visualise different levels of detail. The resolution levels start from a low-resolution overview to higher resolution as a user zooms in on areas of interest. The client and server communicate via WebSockets, and WebRTC is used for peer-to-peer connections when transferring the application's state between devices (e.g., from desktop to VR).

The VRDAVis system was tested through a qualitative study, with participants who were astronomy researchers familiar with Vr technology. The participants were asked to perform various actions using VRDAVis. The tasks involved selecting a file on the laptop, transferring the session to the VR headset, and interacting with the data cube. Key findings were gathered from user feedback, focusing on their experience with the system's usability, interaction with the visualizations, and the overall workflow. Observations on task performance and any difficulties encountered were also collected, along with participants' impressions of working in the VR environment.

The key findings highlighted:

- The client-server architecture supports large data handling by offloading computational tasks to the server.
- Mipmaps enhances system performance by limiting the amount of data the system processes, further reducing computational overhead and improves the ability to explore and visualise datasets progressively.
- The system successfully scales to handle very large data cubes that are several times larger than the memory of the device used for visualisation, making it possible to visualise vast datasets efficiently.
- VRDAVis can function as a remote application, enabling seamless interaction between the client and server over the internet.
- VRDAVis achieves similar functionality when compared to a system like iDaVIE, while iDaVIE uses a co-located computer, VRDAVis is a fully remote system.
- The qualitative user evaluation indicates positive feedback regarding the usability, performance, and experience of VRDAVis.
- The use of three-dimensional visualisation techniques in VR enhances astronomers' ability to interact with and understand the data in ways that two-dimensional screens do not, especially for identifying hidden structures within the datasets.

The VRDAVis system implements a client-side rendering architecture that allows for the visualization of large radio astronomy data cubes on devices with limited computational power, such as laptops and stand-alone VR headsets. This architecture ensures that the data sent to the client is scaled appropriately, preventing performance degradation and reducing risks like simulator sickness. A crucial element of the system's success is its use of pre-processed data. This enables the client to request and visualise smaller, manageable data segments, maintaining overall system performance without overwhelming the device. Data compression is used to minimise the time required to transfer data from the server to the client, reducing interruptions caused by the user waiting for data. The system effectively mitigates the limitations of lower-powered devices by adjusting the amount of data processed to fit within the device's capabilities, ensuring stable client performance even when handling large datasets. VRDAVis has demonstrated its ability to function without the need for high-powered hardware, relying on a remote server to handle computationally intensive tasks. While functionally similar to established systems like iDaVIE. VRDAVis remains experimental and requires further development to refine its features and functionality before it can be fully integrated into research environments.

1 Acknowledgements

I would first like to express my sincere gratitude to my supervisors, Prof. Robert Simmonds and Dr. Angus Comrie, for their guidance, support, and feedback throughout this research project. I would also like to thank the faculty and staff of the Department of Computer Science and the Department of Astronomy at the University of Cape Town for providing a supportive academic environment. I would like to extend special thanks to the Institute for Data Intensive Astronomy (IDIA), whose assistance with supplying computing resources, equipment, and test data was particularly helpful. I acknowledge the support of DARA Big Data, whose financial assistance made this research possible.

2 Introduction

Radio astronomy research involves the processing and exploration of terabytes to Petabytes of data. The quantity of data available for research increases as data collection instruments improve in quality. Modern radio telescopes are equipped with larger and more sophisticated sensors, generating data ranging from terabytes to petabytes each day. The generated data is multi-dimensional and is usually represented as a three-dimensional data cube. To extract knowledge from this data, astronomers need comprehensive visualisations, and analytical tools that enable real-time interaction and exploration. It is a major challenge to visualise such large datasets and involving issues related to data storage, querying, visual presentation, interaction, and personalisation Bikakis (2018).

Astronomy visualisation systems conventionally visualise the three-dimensional datasets using two-dimensional media, such as screens. Ideally, the data should be presented in a way that maintains its dimensional integrity as this creates the most accurate representation of the true nature of the collected data. This allows users to interact with it as if it existed in the real world and mitigates some of the mental load of exploring the data and extracting knowledge. A promising solution lies in the use of VR headsets, which can display three-dimensional data in three-dimensional space. VR headsets use stereoscopic displays and motion tracking to provide a more intuitive and interactive environment for data exploration. However, they are limited by comparatively low computational power comparable to a smartphone and cannot process and visualise the massive data cubes generated by radio telescopes.

An ideal visualisation system must be able to handle very large datasets and operate on machines with limited computational and memory resources such as laptops and virtual reality headsets.

2.1 Objective

This research aims to explore the challenges of astronomy data cube visualisation within an interactive environment as these cubes approach the realm of petabyte in size. It implements a prototype system which can visualise very large volumetric data cubes within a VR environment, by using an intelligent strategy which can scale to handle ever-increasing sizes without sacrificing system performance or user experience. The research will compare the performance of the prototype system to a system with similar functionality, while also taking user feedback into account in a qualitative evaluation. It will evaluate the validity of the chosen system design with the aim of contributing to the understanding of volumetric visualisation tools and their development and implementation.

2.2 Research Questions

The first question investigates whether the prototype visualisation system can be implemented as a remote application. This system design prioritises the user experience, ensuring that the prototype system can be adopted into the research workflow with the least amount disruption to their existing workflow.

The second question examines the system's capability to handle very large data cubes. The system would scale with data cubes that are several times larger than the memory of a lower-powered device. These lowered powered devices are devices such as laptop or VR headset.

Finally, the research will compare the prototype visualisation system to a system which uses a co-located computer, such as iDaVIE Jarrett et al. (2021). This comparison will involve user testing, where participants will evaluate the VRDAVis system and provide feedback on its performance, usability, and overall experience.

1. Can the proposed system architecture be implemented as a remote application for astronomy data visualisation system?
2. Can the system produce performance metrics within certain thresholds when presented with a data cube several times the size of the device's memory?
3. How does the proposed system which uses a standalone virtual reality (VR) headset client, such as the Oculus Quest 2, compare to a system using a co-located computer to take on the majority of the computational workload?

2.3 Overview

The document is structured into five main sections:

1. An introduction that includes the objectives and goals of the research.
2. A literature review which explores existing research, and covers topics such as virtual reality, visual analytics, data visualisation, and graphics.
3. A system design and implementation section describes the construction of the system.
4. An initial evaluation of the system, which assesses the performance of the system.
5. The conclusion summarises the research and outlines areas for improvement.

3 Literature Review

The topics reviewed in this section touch on various fields relating to the development of the VRDAVis system. This involves reviewing other systems which attempt to create real-time and interactive visualisations of large volumetric datasets, as well as other topics which support this process such as analytics, visualisation and rendering strategies. The goal to gather insight into techniques and methodologies that can enrich the VRDAVis system.

3.1 Virtual Reality

VR immerses users in an interactive digital environment using a head-mounted display. The "ultimate display" as described by Ivan Sutherland is a theoretical human-computer interaction device that simulates a reality that is realistic to the point where a person cannot tell the difference between actual reality and the simulated reality Sutherland ([n. d.]). This level of immersion in a completely new world was an appealing idea even before the technological inception of a device that was even remotely capable of achieving it. The virtual world appears realistic to a human through the use of stereoscopic displays and sound. Stereoscopic displays present each eye with a different image for the user to experience simulated depth. Also, gives the user the ability to interact with the virtual objects in a realistic way with tactile feedback. Since the 1960s, the concept of virtual VR headsets has evolved significantly. One of the earliest examples dates back to this era, with Morton Heilig's Telesphere Mask Heilig (1994) which introduced the notion of a head-mounted display. It offered wide vision, a stereoscopic 3D display, and stereo sound; it was later augmented by the introduction of a motion tracking feature in 1961. Over time, various entities, including NASA, Sega, Nintendo, Apple, and Google, have experimented with VR technology. Recent advancements have been marked by Oculus, now owned by Meta, which is releasing affordable VR headsets aimed at mass market consumption, particularly targeting the entertainment sector. However, VR technology is in the process of being adopted in other industries such as education, retail, transportation, energy, consulting, insurance, healthcare, and sports.

VR uses immersive technology to simulate interactive virtual environments where users feel physically present Bowman and McMahan (2007); Wohlgenannt et al. (2020). This is done through the construction of a real or imagined environment, which blocks the real world from the user. It also incorporates interaction methods such as controllers and motion tracking for the user to interact with the environment Brooks (1999); Nash et al. (2000).

VR experience can be evaluated according to three key properties: presence, interactivity, and immersion Walsh and Pawlowski (2002). Presence refers to the feeling

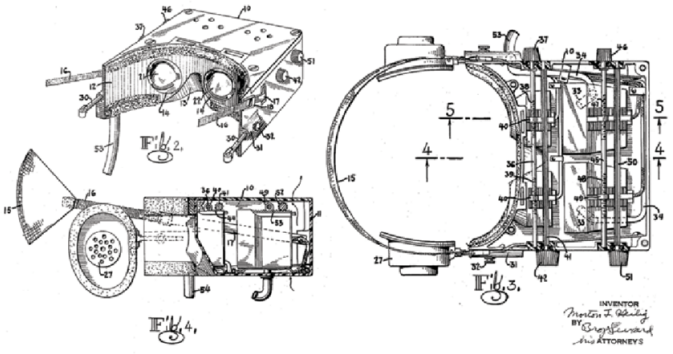

Figure 1: Morton Heilig's patent for the Telesphere Mask Heilig (1994), note the resemblance to contemporary headset design, such as the Meta Oculus Quest and the Apple Vision Pro.

Figure 2: An example of one of the latest iterations of VR technology, Meta's Oculus Quest 3. It can be considered as a extended reality (XR) headset as it can host VR environments while also incorporation augmented reality (AR) features.

of physically being in a place. Within a virtual environment, the presence of the user is simulated by capturing their senses, using visual, audio, and tactile simulations Sanchez-Vives and Slater (2005). The feeling of being present in virtual environments arises from the way that a user perceives and processes information from their surroundings, both in their conscious and subconscious environment Nash et al. (2000). The sense of presence that a user feels is difficult to quantify, as it can be measured in objective and subjective ways. Subjective measurement comes from the level of realism in the virtual environment. As VR technology improves, the level of realism VR headsets are able to display improves with it. The objective measure of presence observes physical indicators in the user; these indicators include heart rate, dilation of the pupils, blink responses, and muscle tension. Some more objective measurements include the extent to which physically reality is excluded from the virtual one, the number of the user's senses that are captured by the virtual reality, the extent of the field of view, the resolution and quality of the displays used, and how the user's body movements match up against their movements in the virtual world Nash et al. (2000); Sanchez-Vives and Slater (2005). Interactions in the real world are naturally as good as they could possibly be according to both subjective and objective measurements of presence. Human senses have specifically adapted for perceiving and interacting in the real world, and therefore it is difficult to simulate the same level of realism digitally.

VR Interactivity refers to how effectively a user can manipulate the virtual environment in real time Steuer (2000). Interaction within a VR environment is usually facilitated through controllers recording gestures and physical movements. In interactive systems, users have agency and control over various elements within the virtual environment, which allows them to make decisions, perform actions, and influence outcomes with their interactions. This can range from simple actions like clicking buttons or navigating menus to more complex interactions such as manipulating objects, and altering parameters. The level of interactivity correlates with the user's sense of immersion and engagement, as it enables them to influence the environment and participate actively in the experience according to their preferences, goals, and intentions. Interactivity is a key factor as it affects the level of presence the user feels within the environment and directly influences the user's sense of presence.

Immersion in VR is more complicated than presence and interactivity. Immersion can be described as subjective involvement, encompassing various dimensions of a user's experience Nilsson et al. (2016). However, there are many facets of immersion. Cognitive immersion manifests when a user engages with the content, experiencing satisfaction and fulfilment upon solving complex problems. Emotional immersion occurs when a user become emotionally invested in narrative structures, experienc-

ing highs and lows which come with a story. Kinesthetic immersion emerges when users receive immediate feedback on their physical actions, which enhances their sense of presence and agency within the virtual environment. Spatial immersion is experienced as users navigate and perform intricate movements, feeling a sense of spatial presence and immersion within the simulated world. These dimensions collectively contribute to a rich and immersive user experience across various interactive platforms and media. The sense of presence relates to the perception of the physical environment and is tightly coupled to how immersed the user feels. Vice versa, the more immersed the user feels in the virtual environment, the easier it is for them to feel present.

While VR systems offer immersive experiences, they face some downsides, notably simulator sickness. This condition is caused by to frame rate drops. Frame rate is the number of images a display such as a screen can cycle through per second. The rapid succession of frames on the display is what causes the image on the screen to appear to have motion. When the frame rate is high, the motion on the display appears smooth; low frame rate makes the motion of the image to appear choppy and unpleasant. It is this choppy movement on the display which leads to a disconnect between user input and display output, which results in feelings of confusion, dizziness, and nausea for some users. Simulator sickness can also occur when the motion cues present in the VR space does not match what the user is physically feeling. The level of susceptibility to simulator sickness varies between individuals, but has some dependence on a users experience with VR headsets as well as their natural tolerance to being in a VR environment. Additionally, as the system's performance declines, the immersive element of the experience diminishes, which highlights the sensitivity of VR experiences to hardware performance. Users experience disconnect from the environment when any aspect of presence, interactivity, or immersion is compromised.

3.2 Visual Analytics

Visual analytics is the practice of analysing datasets using visual representations of the data, these representations are the individual data points of the dataset mapped using a particular visualisation strategy Bikakis (2018).

Visualisations facilitate the emergence of patterns as they are explored by analysts, typically domain experts, enhancing accessibility, interpretation, and meaningful analysis of data Lowe and Matthee (2020); Tukey (1977) through interactive engagement with diverse tools. These visualisations are made with goals in mind, which can be explorative, confirmatory or for presentation Keim (1997). Explorative analysis involves undirected interaction with data which does not have a hypothesis attached

to the exploration. This type of analysis is done to construct a possible hypothesis relating to the data. In contrast to explorative analysis, confirmatory analysis has a more goal-oriented approach; the purpose of the exploration or examination of the data is to confirm or reject a hypothesis. The goal of presentation differs from the goal of a hypothesis where it is only concerned with the technique it uses to present the data to best represent the facts it wishes to communicate. Human analysts are naturally equipped with sophisticated pattern recognition abilities Hassan and Fluke (2011); Taylor (2015); Shneiderman ([n. d.]), and applying these skills while interacting with data visualisations facilitates knowledge extraction Caldarola and Rinaldi (2017); Yi et al. (2007); Becker et al. (1987); Glueck et al. (2014); Bikakis (2018); Muhlbacher et al. (2014); Shneiderman (2008). Large datasets within the Big Data field require analysts' cognitive abilities to make sense of the data, using perceptual grouping, image segmentation, and object recognition Lowe and Matthee (2020). These skills allow elements such as clustering, outliers, patterns, and correlations become apparent. A visualisation can communicate the essence of a dataset quickly and effectively regardless of size Fisher et al. (2012). The aim of visual analytics is to support the analysts' analytical reasoning and research with interactive visualisations of datasets Tufte (1983); Ali et al. (2016); Becker et al. (1987).

For the users to explore the visualised datasets there must be some interactions they can perform to affect the representation Yi et al. (2007). Without interaction the representation would be a static image Becker et al. (1987). Static images do offer value in visual analytics, but this value is limited once the dataset exceeds a certain size and the granularity of the data becomes too fine. Interactions performed on the representation must produce an instantaneous change in the visualisation to maintain the user's attention. There is a threshold of 10ms or less between interaction input and response to facilitate an uninterrupted dialogue between the user and the data Glueck et al. (2014); Li et al. (2016); MacKenzie and Ware (1993); Muhlbacher et al. (2014).

3.2.1 Progressive visualisation

As a user analyses a representation of the dataset, the visualisation should allow the user to steer the appearance of the visualisation through interaction and should also allow them to drill down into areas of interest within the data. This is the process of progressive visualisation where initially a low fidelity representation is displayed and as it is explored, higher fidelity data is brought into the visualisation Zhao et al. (2017); Ahrens et al. (2005); Rosenbaum and Schumann (2009); Li et al. (2016); Tufte (1983). The user steers the visualisation's progression through the data and the visualisation changes as they explore the data Zhao et al. (2017); Muhlbacher et al. (2014); Fisher et al. (2012). The data stored at different levels of

fidelity are pre-processed into a hierarchical order so that required data can be found easily Glueck et al. (2014). Pre-processing data used for visualisation can dramatically improve the user experience Shneiderman (2008). Some of the visualisation tools for interactions should include scaling, translation, rotation, filter, overview, and detail-on-demand Zhao et al. (2017); Becker et al. (1987); Shneiderman ([n. d.]); Keim (1997); Baracaglia and Vogt (2019); Ferrand et al. (2016).

Scaling data correctly is important for lower-powered devices such as the VR headset and personal laptop computers, as they have comparatively fewer computational resources available. The visualisation of points which translates to a size which is smaller than a single pixel on a screen is a waste of resources and should be minimised Li et al. (2016); Ertl et al. (1999).

3.2.2 Virtual reality in visual analytics

Human eyes are naturally suited for three-dimensional space and are more in accustomed to interacting with three-dimensional objects and VR simulates this interaction Ferrand and Warren (2018); Abidi et al. (2017). VR headsets provide an inexpensive and portable environment for multi-dimensional visualisations, which provides the ability to interact with the visualisation. Three-dimensional data visualised in three-dimensional space grant a holistic view of the of the data Ferrand et al. (2016); Farr et al. (2009). An effective visualisation bridges the gap between the data contained in datasets and the human ability to extract knowledge. The stereoscopic environment produced by a VR headset such as the Oculus Quest 2 allows the user to see the data with a perception closer to a real-world environment, fostering user friendly interaction which resembles interaction in the real world.

3.3 Data Visualisation

This section will summaries the different techniques and strategies used to create visualisations of radio astronomy data cubes or any volumetric data. Volumetric datasets can be thought of as data cubes as they have data along the x-,y- and z-axes. They are commonplace in fields such as astronomy, medicine, oceanography, molecular modelling, and engineering. Data cubes can be described as a set of samples *(x, y, z, v)*, where the value of v is some property of data at the three-dimensional coordinates *(x, y, z)* Kaufman (2000). A point within a data cube is a three-dimensional pixel and is referred to as a voxel or volume pixel Kaufman (1996). In astronomy datasets x and y are used for spatial information where the third axis, z, is usually frequency, not distance. The steps through the z-axis are refereed to as channels and the number of channels on the z-axis depends on the velocity resolution and bandwidth of the receiver Taylor (2015). The data is a

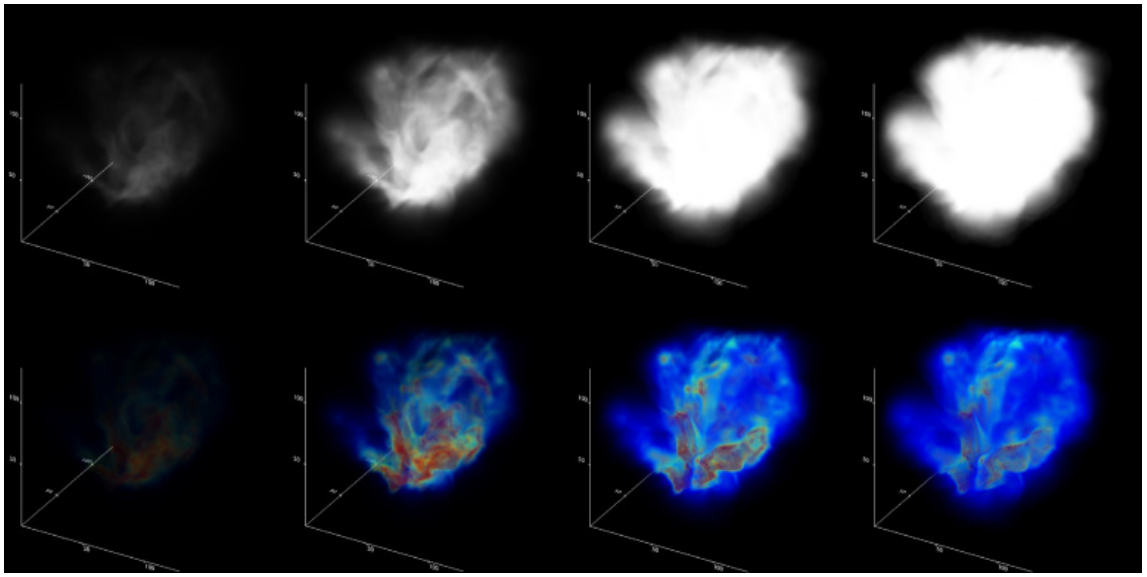

Figure 3: An astronomy dataset visualisation generated by Frelled Taylor (2015) depicting how the addition of colour and opacity applied to the voxels effectively exposes hidden structures within the dataset. It shows how various ranges of opacity affect the visualisation but also how colour adds to its readability.

representation of a measurable property of astrophysical phenomenon, and can be represented as binary values or values within a range.

Data visualisation is the presentation of data as a static or interactive image Bikakis (2018). Visualisation is imperative to making sense of abstract datasets such as those prevalent in radio astronomy research Glueck et al. (2014); Yang et al. (2017), and is important for the exploration of data and the communication of findings Ferrand and Warren (2018). Visualisations allow enormous amounts of quantitative data to be shown in a limited space, while also maintaining the message contained within the dataset Fisher et al. (2012). Data visualisation has some distinct objectives Norris (1994); McReynolds and Blythe (2005); Hassan and Fluke (2011):

- Allow the user to explore the data and obtain a deeper understanding Bikakis (2018).
- Expose features within the dataset which would otherwise have remained hidden.
- Obtain quantitative results from the data, and communicate the results to others.

There are certain characteristics present in the data which make visualisation dif-

ficult Hassan and Fluke (2011). The datasets produced by astronomy data gathering instruments produce massive amounts of data, giga-bytes to peta-bytes, large enough to fill the memory of multiple conventional desktop computers. This makes the datasets difficult to manage and process for visualisation. There is no dominant file type used to represent astronomy data, which makes the development of a generic visualisation tool difficult. Conventionally FITS files Wells and Greisen (1979) have been used to represent data cubes. The cubes are divided up into slices along the z-axis. Each of the these slices can be visualised as an image consisting of pixels with an x and y coordinate. A user can move through slices of a data cube sequentially and can be played through like frames in a movie. A major downside of visualising the data in this manner is that it does not let the user get an impression of how the dataset looks as a whole. Therefore, limiting how quickly and effectively a user can construct a mental model and begin extracting insight Norris (1994). That data also has a high dynamic range and a low signal-to-noise ratio, which makes distinguishing genuine sources between large portions of noise difficult for astronomers.

3.3.1 Visualisation Techniques

There are different visualisation techniques which can be used to produce a representation of the dataset. However, the nature of the data dictates the appropriate visualisation method to be used Hassan and Fluke (2011). The common techniques used for generating comprehensive visualisations of a volume dataset are points, splats, iso-surfaces, and volume rendering.

Points Each point in a dataset represented as a fixed-width pixel to a picture plane as if it were a three-dimensional scatter-plot. While it is the most straightforward technique for representing data, this approach is limited by the available resolution or pixel density of a display. This is not a suitable method for visualising astronomical data. Simply plotting the data as a three-dimensional scatter plot makes it difficult to derive meaning from the data visualisation. For example, the top left quadrant of Figure 4 when compared to the bottom right quadrant is much less semantically comprehensible.

Splats This visualisation technique uses small two-dimensional textures to represent data points, and uses a method known as "bill-boarding" Taylor (2015), to ensure the textures always point towards the camera regardless of the scene's orientation. When many splats are combined, this produces an effect that is much like volume rendering, but without the computational overhead of ray-tracing.

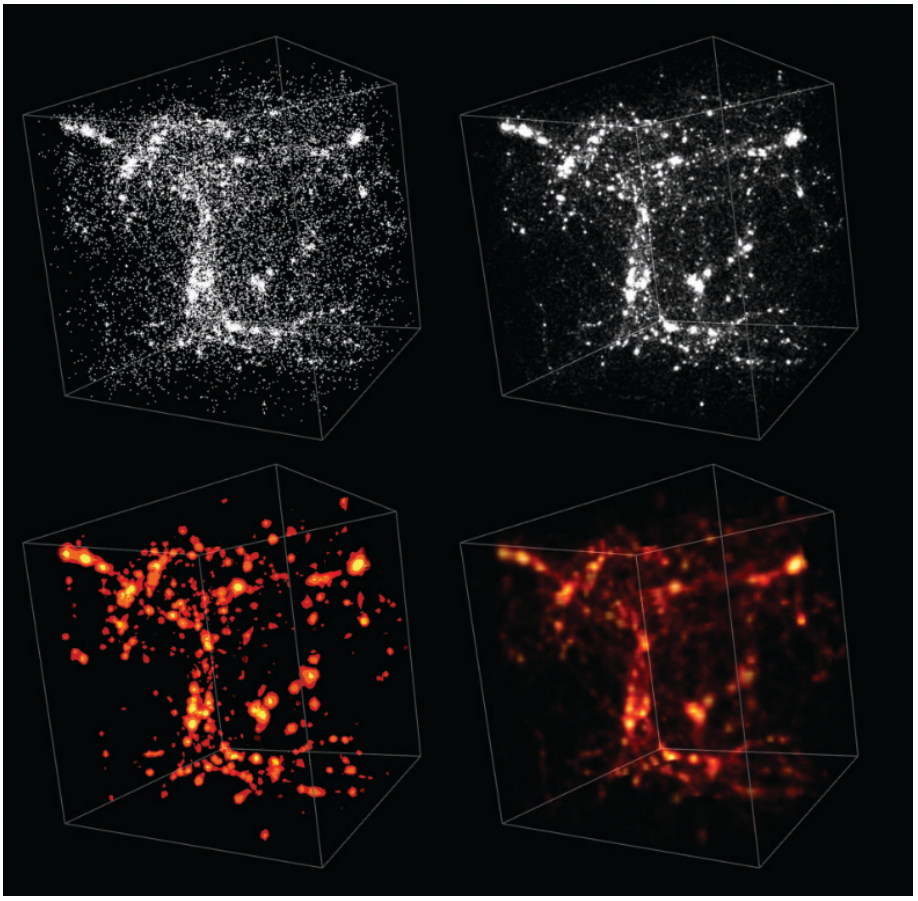

Figure 4: Four visualisations of an astronomy dataset using the common visualisation techniques Hassan and Fluke (2011), clearly depicting how the different visualisation techniques affect the presentation of the data. Point (top-left), splats (top-right), iso-surface (bottom-left), and volume rendering (bottom-right).

Iso-surface Polygon-based rendering, also referred to as surface rendering, is used to visualise volume data using polygons. The majority of three-dimensional models use this method to represent their forms. Surfaces or polygons must be extracted from volume data before it can be visualised. Part of the visualisation process involves determining where the boundaries of the objects are and defining a skin for each region. There are various methods for calculating these boundaries, such as: marching cubes, marching tetrahedra, multi-resolution iso-surface extraction, and surface wave propagation. Iso-surface rendering is commonly used to search for correlations between different scalar values, but is not used as a visualisation method because the visualisation is not comprehensive enough for knowledge extraction.

Volume Rendering This technique renders each data point as three-dimensional pixels, also referred to as voxels. Voxels are arranged on the x-, y- and, z-axes to construct a three-dimensional image. To make these structures more comprehensible, a colour transfer function is used to assign an opacity and colour to each voxel. Volume rendering uses ray-tracing or ray-marching shaders to render the visualisation. These types of shaders use significantly more computing resources compared to the traditional shading model Wittenbrink et al. (2000). There are several optimisations and acceleration techniques that can tailor the shader to data being visualised. However, it is a favoured method for representing volume data as it produces a complete view of the external surface as well as the internal three-dimensional structures within the dataset.

3.3.2 Two-dimensional versus Three-dimensional Visualisations

Three-dimensional visualisation techniques are better-suited for representing volume datasets, compared to two-dimensional visualisation techniques.

Some intuitive understanding of the data is missed when the data is examined using two-dimensional approaches Kent (2013). These approaches examine pieces of the data, such as individual slices from the dataset, or are played through sequentially like a movie. This method of examination does not allow the user to observe the dataset as a whole. Three-dimensional visualisations highlight hidden features within the data that would otherwise have remained hidden Ferrand et al. (2016). Comprehensive visualisations are important for communicating results quantitatively. Therefore, visualisation which represent the data in a more comprehensive way is preferred.

3.4 Rendering

The visualisation of large amounts of data becomes problematic when processing the data consumes all of the computational resources of a non-specialised system. This is because the resource requirements for processing increase with the size of the data-sets Shneiderman (2008). Visualisations of large data cubes can still be produced by systems with specialised hardware, but these systems are often limited by their connection to a single geographical location and restricts the number of users able to access the system at a given point in time. Visualisation of data is essential to acquire an understanding of it Yang et al. (2017). Additionally, making the visualisation interactive increases its effectiveness. The addition of interaction increases computational overhead as the dataset being visualised must be rendered for each frame displayed. Changes in the visualisation's appearance resulting from interactions should be perceived as instantaneous. Developing a system for produc-

ing interactive visualisations of gigantic datasets in real-time is a challenge because hardware will always be pushed to its limits. Waiting for hardware technology to catch up with the size of the datasets is futile, as the datasets also increase in size at a steady rate over time. The increasing size of the datasets can be attributed to many things, but specifically in astronomy it stems from data-capturing instruments becoming more sensitive and more accurate. To solve this problem would require a system which can scale consistent performance with the ever-increasing datasets without relying on the brute-force strategy for producing visualisations Ali et al. (2016). In the brute force strategy, every single data point within the dataset is visualised regardless of the size of the dataset.

While visualising the entire dataset produces a highly accurate depiction of the data, the amount of time it takes to produce the visualisation is directly linked to the computational power of the hardware. It could potentially take a long period of time to produce the visualisation if the visualisation is produced on lower-powered hardware Fisher et al. (2012). The length of time can be affected by hardware issues such as system bandwidth, which depends on factors such as the graphical computations, the speed of the display hardware, and the speed at which information can be passed between components Becker et al. (1987). All of the factors present bottlenecks where system performance is potentially stunted.

The rendering speed of the visualisation is crucial to user experience MacKenzie and Ware (1993), especially in the context of VR, where if the time between updates takes too long it could cause simulator sickness in the user, making the system unusable from the user's perspective. For devices with lower-powered hardware to be able to process and visualise the data without negatively affecting system performance the dataset must be reduced in size to reduce the amount of time required to produce a visualisation. This can be achieved either through breaking the dataset up into smaller pieces Masiane et al. (2019); Li et al. (2016) or using some sort of data reduction technique to reduce the overall resolution of the dataset through downsampling techniques such as filtering, clustering and sampling Masiane et al. (2019); Glueck et al. (2014).

There are problems with both of these approaches. Dividing the data cube into subsections has the potential to affect the integrity of the data and obscure the context of a section within the larger dataset Abidi et al. (2017). Downsampled data does attempt to preserve as much detail as possible Dumitrescu and c. Boiangiu (2019), although the integrity of the data cube is affected and some data can be lost. If the time reduction provided by downsampling the data obscures potential insights, the visualisation becomes useless. Both of these techniques reduce the accuracy of the visualisation.

The accuracy with which the visualisation is depicted is incredibly important to the analysis of the data, as discussed further in Section 3.2. It is advantageous to give the user a high-level overview of the data at the start of their workflowLi et al. (2016); Lowe and Matthee (2020); Zhao et al. (2017). This overview can be a downsampled version of the data, which would reduce the processing time because there is less data to be visualised overallMasiane et al. (2019). The user must be able to dig deeper into certain areas of interest in order to explore the dataset and uncover insights. In the architecture of the astronomy visualisation system iDaVIE Marchetti et al. (2020), the user is initially presented with a downsampled version of a data cube and can then explore smaller sections at higher levels of resolution. iDaVIE was made for the exploration of data cubes on a VR device, but the majority of the computation is performed on a co-located computer, typically with mid-range computational power and a GPU. The iDaVIE system workflow is as follows, the user explores the initial data cube and finds an area of interest they would like to explore further. The user selects that area and crops the section. iDaVIE fetches the higher-resolution version of the selected portion of the data cube. The user sees these intermediate results at the early stage of the data processing, which allows them to do further exploration based on current knowledge. This method of progressive visualisation facilitates the exploration of large data cubes. It allows for speedy access to a visualisation and minimises processing time Zhao et al. (2017) while also maintaining the accuracy and integrity of the data Masiane et al. (2019). This method provides required data on demand but does not overwhelm the user with too much information at once. Even if it is a subsection of the larger data cube the context of the piece of data is understood. The user understands how that particular section relates to the whole data cube. This rendering strategy supports the user workflow Hassan and Fluke (2011), initially presenting the user with the full-resolution visualisation does not have a significant improvement to knowledge extraction as, the level of detail visible on the display is limited by the display's resolution. The visualisation wastes computational resources by attempting to render points which correspond to an area smaller than a pixel on the chosen display hardware.

3.4.1 Remote Rendering

The remote rendering strategy is an approach to rendering which offloads computational overhead from lower-powered hardware, such as a laptop or personal computer, to higher-powered hardware, such as a dedicated server. It makes use of the client-server model; the server component stores the large data-sets and performs the computationally expensive processes, whereas the client provides input as requests to the server and displays the output of these requests to the user. This makes it possible to produce visualisations of datasets which are larger than the client's resources can accommodate Hassan and Fluke (2011). For remote rendering to func-

tion as intended, the client must be able to access any portion of the dataset, and the server and client systems must coordinate with each other to render and display the visualisations in real time Glueck et al. (2014). There are also bottlenecks present in this approach that must be considered. The largest bottleneck is the amount of data that can be sent through a network channel in a period of time Abidi et al. (2017). To mitigate the bottleneck the amount of data transferred between the server and client systems needs to be minimised. Compression is the most common method for reducing the bottleneck. However, compression and decompression consumes CPU on both the client and the server systems and adds time to the rendering process. It also presents the added chance of losing data during the compression-decompression process. The system would use lossy or lossless compression, depending on the type of data and how important the integrity of the data is to the functionality of the system.

There are two rendering strategies present in remote rendering; server-side and client side rendering, both of which are prevalent in web application development. Systems choosing an appropriate approach for their system architecture must consider which strategy would compliment the functionality of the system Iskandar et al. (2020).

Client-side Rendering In client-side rendering, data is downloaded from a server and then rendered in the client browser. More specifically, in a web application a request is made to a server and a bundle of data is downloaded. Once the bundle reaches the web application it extracts the data from the bundle and uses the data in application. These types of applications use API calls or WebSocket connections to request data from servers. The process of rendering of volumetric data is the same, data is stored on the server, the data is bundled and sent to the client, the data is extracted and then rendered to produce a visualisation on the client system. This reduces processing time by rendering the data on the client system. The compression and decompression of the data during the transfer process can affect its accuracy. The integrity of the data could also be affected by the downsampling process it went through on the server.. The amount of data that can be rendered is limited to the computational power of the client system while the client system performs other tasks besides those related to the rendering of the data visualisation Becker et al. (1987). In a comparison between client-side and server-side rendering, client-side produces a better user experience Iskandar et al. (2020), as the interactions performed on the client are perceived as instantaneous.

Server-side Rendering Server-side rendering involves storing the data on the server and also rendering the data into a visualisation on the server. Frames of this visualisation are sent to be displayed on the client web application. The client system has no part in the rendering process, it just displays the frames it receives

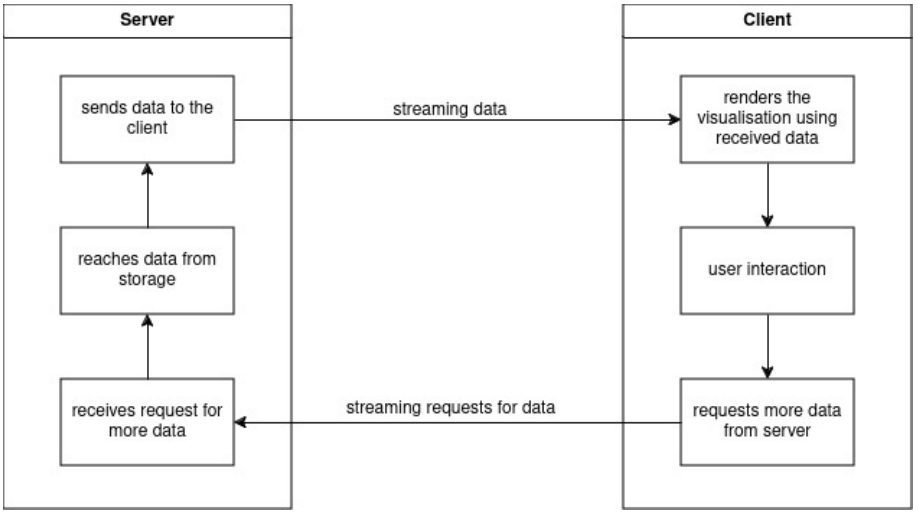

Figure 5: An example of the process of client-side rendering where data is sent from the server to the client to be rendered. The user interacts with the data on the client and only requests more data from the server when it is required.

from the server. The user can manipulate the visualisation by forwarding input capture by the client application to the server, following the classic request-response model for web applications. The server renders the three-dimensional visualisation of the data but streams two-dimensional images to the client. This strategy increases the accuracy as the integrity of the data is unaffected as it is not processed or transferred, however, it does increase processing time. The main point where this processing time becomes apparent is when the user performs interactions with a visualisation on the client system. The user must wait for the interaction to be forwarded to the server, then the visualisation must be re-rendered, and finally the updated frames are streamed to the client. The time in which these tasks are done increases if the frames are also compressed for transfer over a network, and subsequently decompressed on the client. The speed of the network the frames are streamed on also affects the processing time of the feedback to the user on the client system. The slower the network, the more time the user has to spend waiting for feedback. The server could also have a delayed response because it is performing other tasks, and handling many clients' requests Becker et al. (1987).

3.5 Current Visualisation Systems

Extremely large datasets are commonplace within radio astronomy research. As this research progresses, the datasets are predicted to balloon in size even more at ever faster rates. Advances in astronomical technology, such as larger detectors,

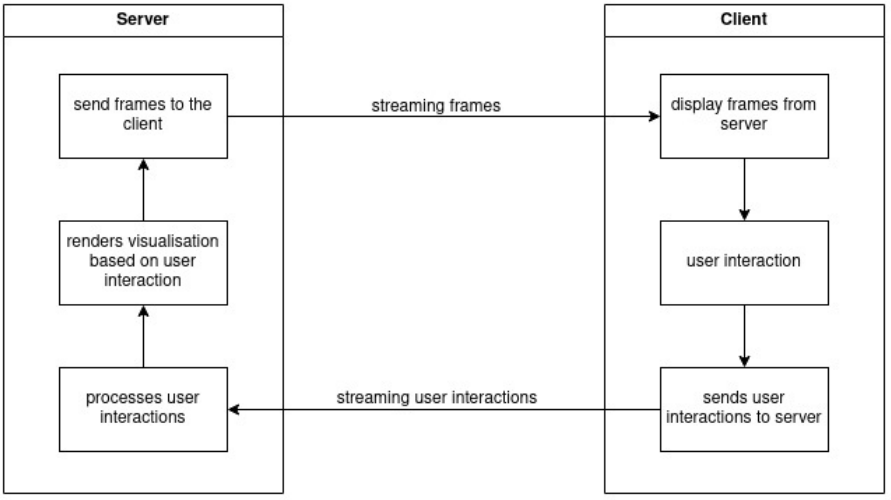

Figure 6: An example of the server-side rendering process, where the rendering of the data is done on the server and the frames are streamed to the client. The user interacts with the visualisation on the client but the interactions are sent to the server. The rendering is updated based on the user interactions and the frames are once again streamed to the client.

increases in bandwidth, and astrophysical simulations, drive the need for visualisation solutions. Specialised systems have been developed to cater to this aspect of astronomy data, as well as to astronomy researchers. While not all of these systems attempt to apply a generalist approach to volumetric data visualisation, important points can be taken from the systems reviewed. The system architectures and strategies employed to handle the enormous amount of data produced by modern radio astronomy data-capturing instruments emphasise points that can be applied to various systems, as there are some shared goals. Sources Ferrand et al. (2016); Hassan and Fluke (2011); Naiman (2016); Kent (2013); Shneiderman (2008) highlight a need for a generic standalone, cross-platform application capable of processing and visualising large amounts of volumetric data. These visualisations, used in addition to traditional statistical visualisations, have complementary effects on knowledge extraction by communicating information clearly and effectively Caldarola and Rinaldi (2017); Goodman (2012); Naiman (2016); Rosenblum et al. (1994) especially as the variety of data grows and becomes more complex.

3.5.1 Big Data Visualisation Systems

While not specifically geared towards astronomy data visualisation, these systems face similar issues handling large amounts of scientific data, and reviewing some of

these systems could add valuable insight which can be applied to the development of radio astronomy visualisation systems.

Big data as a term can be defined as "datasets whose size is beyond the ability of software tools to capture, store, manage, and process" Manyika et al. (2011); Shneiderman ([n. d.]) or "data which exceeds the capabilities of commonly used hardware environments and software tools to capture, manage, and process it within a reasonable amount of time" Merv (2011). These datasets have characteristics relating to the three V's of *big data*: volume, velocity, and variety. As volume increases, problems involving processing, storing, and extracting knowledge arise. When variety increases it complicates the storage and analysis of the data because of the different structures of data being stored. Velocity relates to the speed at which data is generated, this has the potential to compound the issues of storing and processing a high volume of data Caldarola and Rinaldi (2017). This is because processing data takes time and if data is not process fast enough it creates a backlog when new data is inbound.

There are various software packages and platforms to assist the user in making sense out of massive amount of information, each tailored to a specific use case. For big data visualisation and analytics, great consideration must be put into the method of visualisation because the structure of the data affects the resulting visualisation. Generally, the tools value flexibility and user convenience. By making these tools available on multiple platforms for ease of access and by having them be programming language agnostic, allowing the user to customise their workspace.

A general solution to visualisation and analytics is complicated because it is impossible to account for all methods of visualisation and for the variety of data. But most are aimed towards business analytics. Some examples of data visualisation software for specific data types are:

- Timeline - specialised to create interactive timelines from a spreadsheet
- Commetrix and Cuttlefish - used for dynamic network visualisation and analysis
- Cytoscape - used for visualising molecular interaction networks and biological pathways, originally designed for biological research but is now also used for complex network visualisation and network analysis

Current systems used for exploration and visualisation cannot handle the size of many contemporary datasets and restrict themselves to analysing only comparatively small datasets. This is mainly due to them using a brute force method of visualising the entire dataset regardless of how it will appear on a display Bikakis (2018).

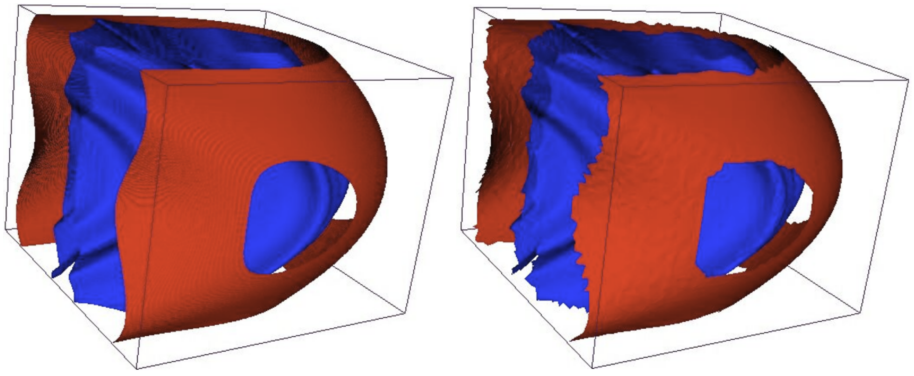

Figure 7: An example of a level-of-detail technique specified by ParaView. The model on the left is rendered at a higher resolution than the one on the right. While the quality of rendering is impacted, but improving rendering performance by having a lower resolution model to visualise.

The free-to-use game engine Unity Marchetti et al. (2020); Ferrand et al. (2016); Ferrand and Warren (2018) and the open source three-dimensional modelling and animation software Blender Taylor (2015, 2016); Kent (2013); Naiman (2016) have been used for fast prototyping of various astronomy visualisation systems. Blender is favoured for three-dimensional rendering capabilities and it three-dimensional space manipulation tools, particularly useful for generating animations and graphics with astronomy data.

ParaView ParaView Ahrens et al. (2005), is an example of a tool created for scientists to visualise and analyse extremely large datasets, while emphasising the workflows of the domain experts. It highlights speed and performance as key aspects of the software's functionality. Its rendering strategy involves executing programs in parallel on either shared or distributed memory resources. It can achieve high frame rates suitable for interaction by utilising level-of-detail techniques. These level-of-detail techniques, also known as progressive visualisation, involve storing the dataset at many different resolution levels, which are also know as mipmaps. They reduce the amount of data that has to be rendered, which also impacts the quality of the visualisation. The dataset being visualisation must find a balance between the resolution of the data and the performance of the system.

ParaView uses a layered architecture Ahrens et al. (2005). The VTK foundation layer handles the data representation and algorithms Ahrens et al. (2005). The second layer handles the parallel extensions to the visualisation toolkits, while supporting streaming of many data types and parallel execution. The third and final layer is ParaView itself, which provides a graphical user interface and the visualisa-

tions.

3.5.2 Astronomy Data Visualisation Systems

Among the systems used to process and visualise volumetric datasets there are some have implemented solutions exclusively for astronomy datasets.

One of the first systems which implemented a working solution to visualise very large astronomy datasets was an unnamed framework that leveraged a cluster of 96 GPUs to produce interactive and real-time visualisations Hassan et al. (2013). The creation of this framework anticipated the challenges that astronomy visualisation systems would have to overcome in the near future. It utilised a GPU cluster facility as the main computational force for a server, which allowed users to interact with framework through a remote client with an appropriate user-interface. It's strategy to avoid slow read and write operations from short-term and long-term storage is to load the entire terabyte dataset onto the GPU cluster, referred to as an "in-core solution". This may seem extreme, but this framework was intended to be hosted by a supercomputing facility where these resources are available. While also aiming fora frame-rate between 5 and 10 frames per second.

CARTA The acronym CARTA stands for Cube Analysis and Rendering Tool for Astronomy Comrie et al. (2021). Its objective is to provide a usable and scalable system using modern web technologies, and its main hallmark is being able to load and render a dataset of several terabytes in a matter of seconds. It achieves this by storing the dataset at various levels of resolution and sending only the requested data required to produce a comprehensible rendering, instead of attempting to render the whole dataset at once. It makes use of the client-side remote rendering approach for handling large datasets, by hosting them on a remote server. A local web-based client accesses the data through a network connection, the data is transferred to the client and rendered within the web browser. It is designed for the ALMA, VLA, and SKA pathfinders and works with data types commonly used in astronomy research such as FITS, CASA, MIRIAD, and HDF5.

iDaVIE This software for data cube exploration within a virtual environment, providing tools to navigate and interact with the visualisations. Functionality to view metrics and interactively mask regions of data in the VR environment to catalogue sources is particularly useful. The VR environment integration enables a unique and immersive perspective of the data, affording intuitive interactions. It was created using the Unity game engine and uses the client-server architecture within its design Marchetti et al. (2020). It stores the datasets on the server but also renders the visualisation using the server. A prototype system similar to iDaVIE created for

public outreach Ferrand and Warren (2018), is a tethered standalone system which means that the server and the client are connected to each other with a physical connector such as a cable. It also uses Unity with custom scripts to visualise FITS files and displays the visualisation in a VR environment using the HTC Vive. It can produce iso-surface or volumetric visualisations of a dataset.

Frelled A general-purpose astronomy viewer Taylor (2015) which uses a set of Python scripts to create three-dimensional visualisations of FITS files inside Blender, and is used for visual source extraction, cataloguing, and analysis. It can visualise large datasets of roughly 600,000 points at more than 10 frames per second, while providing tools to mask data interactively, functionality similar to iDaVIE. The FITS file can be explored in a three-dimensional space, instead of two-dimensional slices. It provides a significant speedup in source cataloguing compared to other viewers, facilitating the recording of hundreds of sources a day. Visual source extraction by an astronomer is still currently more effective than automated processes.

AstroBlend A Python package for Blender Naiman (2016), combining the three-dimensional capabilities of Blender with astronomy analysis tools. Allows astronomers to analyse data alongside visualisations, while simplifying the importing process.

SlicerAstro A multi-platform open-source package for visualising and processing medical images, which is an extension of the open-source software 3D slicer Fedorov et al. (2012); Punzo et al. (2017). It aim is to provide an interactive three-dimensional visual analytics tool using two-dimensional displays and interaction devices. It is implemented using C++. Its features include interactive filtering, three-dimensional masking and modelling, and support for FITS files.

E0102-VR An experimental standalone application using a VR display and interaction devices. Its aim is to demonstrate the benefits of using human depth perception for fast and accurate characterisation of three-dimensional structures Baracaglia and Vogt (2019). It explores the scientific potential of VR for displaying multi-dimensional datasets in astronomy and astrophysics. The system pre-processes datasets into an iso-surface representation as it does not currently support volume rendering. These pre-processed datasets can be swapped out with no effect on the functionality, but the system is currently coupled to a single dataset (SNR E0102) for experimental purposes. The size of the dataset is limited because all the computation is done on a single computer.

KARMA A toolkit to ease development of visualisation applications Gooch (1996) and is a suitable environment for creating "production" tools catering to file formats

commonly used for radio astronomy data. It supports file types such as AIPS (Astronomical Image Processing System), AIPS++ (Astronomical Information Processing System), DRAO (Dominion Radio Astronomy Observatory), FITS, GIF, GIPSY, IRAF, Karma, Miriad, Starlink Image, Sun Rasterfile, Targa TruVision, and TIFF. While also supporting tools relating integrated communication, integrated graphics, intelligent arrays which are multi-dimensional, dynamically allocated at runtime, which know their own size and type. As well as offering programmes for volume rendering of data cubes, play movies of data cubes, inspecting multiple images and cubes at the same time, slice a cube,superimpose images, interactive position-velocity slices, look for expanding shells, interactive co-ordinate placement, and rectangular to polar gridding of images Gooch (2006). The toolkit has uses a layered architecture. Each layer provides one or more packages where each package encapsulates a functional concept, such as interprocess communication or a graphics canvas. Every package has a class and uses member functions to expose internal mechanisms. Packages at the same levels or higher levels have no dependencies between each other

DS9 Also known as SAOImageDS9 is a stand-alone application for astronomical data visualisation Joye et al. (2025), which supports FITS file format, binary tables and, frame buffers. It offers features such as region manipulation, various scale algorithms, and colourmaps. As well as advanced features like two-dimensional, three-dimensional, and RGB frames, mosaics, tiling, zoom, crop, rotate, and pan.

3.6 Findings

Virtual reality technology has made significant strides since its conceptualisation, evolving into immersive experiences that blur the lines between real and virtual environments. Central to the effectiveness of VR experiences are the concepts of presence, interactivity, and immersion. Presence is the sensation of physically existing within the virtual world, influenced both by subjective perceptions and objective indicators. Interactivity allows users to engage dynamically with the virtual environment, enhancing their sense of immersion and agency. Immersion, with its cognitive, emotional, kinesthetic, and spatial dimensions, contributes to a rich user experience across various platforms. Challenges such as simulator sickness highlight the importance of high-performance hardware in maintaining immersion and preventing adverse effects.

The integration of visual analytics into data exploration and analysis practices offers a dynamic approach to uncovering insights and patterns within datasets. Through visual representations, analysts are empowered to interact with data, leveraging their inherent pattern recognition abilities to extract knowledge effectively. The goal of

visual analytics is to enhance analytical reasoning and research through interactive visualisations. The efficacy of visual analytics hinges on user interaction, where progressive visualisation plays a pivotal role. It benefits both the flow of exploration for the user as well as the overall functionality of the system. By allowing users to steer the representation's progress through data exploration, it facilitates a seamless transition from low-fidelity overviews to detailed insights. This iterative process, supported by techniques such as zooming, filtering, and immersive experiences like VR, enables analysts to navigate complex datasets efficiently. VR offers a compelling avenue for immersive analytics, providing an intuitive environment for multi-dimensional visualisations. By bridging the gap between quantitative data and human perception, VR not only enhances interaction but also enriches the understanding of complex phenomena, fostering a more holistic and user-centric approach to data analysis.

The data visualisation section 3.3 also discusses the benefits of of using a three-dimensional visualisation rather than two-dimensional views. The visualisation of volumetric data cubes, particularly in fields like astronomy, presents a challenge because the complexity and scale of the datasets involved. These data cubes are essential for understanding phenomena in various scientific disciplines, including astronomy, medicine, oceanography, molecular-modeling, and engineering. Visual representation techniques unlock insights from these datasets.

One of the key objectives of data visualisation is to enable users to explore and understand complex data more deeply while having the means to communicate their findings to others. However, the unique characteristics of radio astronomy data, such as its massive size and dynamic range, present specific challenges for visualisation.

Various visualisation techniques exist, each with its own advantages and limitations. These include points, splats, iso-surfaces, and volume rendering. Among these, volume rendering emerges as the most comprehensive method for representing volume data, providing a detailed view and of external surfaces and internal structures within the dataset. It is resource-intensive, volume rendering offers a robust approach for extracting knowledge from volumetric datasets.

Furthermore, the choice between two-dimensional and three-dimensional visualisations is crucial. While two-dimensional approaches may be enough for certain analyses, three-dimensional visualisations are generally more effective for representing volumetric datasets comprehensively. They reveal hidden features and allow for a more intuitive understanding of the data, enhancing communication of quantitative results.

In summary, effective visualisation techniques are unlocking insights from volumetric datasets in fields such as astronomy. By leveraging advanced visualisation methods

like volume rendering and prioritising three-dimensional representations, researchers can better understand complex data and communicate their findings more effectively.

The rendering strategies section addresses the challenges of visualising large datasets and the strategies that balance computational efficiency with maintaining data integrity and user experience. Traditional approaches, such as brute force visualisation, strain computational resources and impede real-time interaction. To overcome these limitations, progressive visualisation techniques, used by iDaVIE, offer a solution. By presenting users with initial down-sampled overviews and enabling them to explore areas of interest in greater detail, these methods optimise processing time without sacrificing accuracy or overwhelming users with excessive information. Moreover, remote rendering strategies, whether server-side or client-side, offer avenues for offloading computational overhead and accommodating datasets larger than the client's resources can handle. However, network bottlenecks and trade-offs between data compression and integrity must be managed carefully to ensure efficient data transfer and timely user feedback. As datasets continue to grow in size, the development of scalable visualisation systems that prioritise both performance and usability remains imperative.

The review of existing systems has highlighted several key points. For instance, the advantages of pre-processing large datasets, where systems retrieve higher-detail sections of data to expedite data visualisation. Three-dimensional representations enhance the viewing experience for multi-dimensional datasets, and VR displays offer a more intuitive interface compared to two-dimensional displays. VR technology capitalises on user depth perception and spatial awareness, further improving the visualisation process.

Efforts have been made to utilise datasets in the FITS file format to generate 3D visualisations, in system such as *Frelled* and *SlicerAstro* While these implementations yield effective three-dimensional visualisations, the FITS file format is not inherently conducive to efficient visualisation because of its storage of data in sequential slices, like pages in a book. This sequential storage method makes traversing and extracting data for three-dimensional visualisation a cumbersome task. Additionally, files may be too large to visualise in their entirety, necessitating the division of datasets into more manageable pieces. However, breaking datasets into fragments risks obscuring the context of individual pieces and potentially diluting the insights the dataset would otherwise convey as a whole.

During the development of the experimental system E0102-VR, valuable insights were gained regarding the utilisation of VR for viewing and interacting with multi-dimensional datasets. The depth perception afforded by VR aids in comprehending complex three-dimensional structures and facilitates rapid measurement of distances

and angles in three dimensions.

These developments underscore the transformative potential of virtual reality in scientific data visualisation, particularly for exploring large, complex, multi-dimensional datasets. While challenges remain, ranging from data format limitations to computational constraints. Emerging strategies such as progressive rendering, remote visualisation, and intuitive VR interfaces are actively addressing these shortcomings. By harnessing the unique strengths of VR, enhanced depth perception, spatial interaction, and immersive engagement. Researchers are able to navigate intricate datasets with greater clarity and communicate findings more effectively. As tools and techniques continue to evolve, the integration of VR into data analysis workflows promises to reshape how we explore, interpret, and share knowledge across a broad spectrum of scientific domains.

4 Technologies Employed

To construct the prototype system the language of implementation had to be considered to ensure they align with the goals set out for this prototype. Various libraries had to be considered and selected. Libraries for three-dimensional graphics, VR embedded environment, web-Socket connections, peer-to-peer connections, and data transfer. With the creation of packages easily integrated using package managers, there are many to choose from with varying degrees of functionality, documentation, community support, and how well it would fit with the other packages already used. When selecting one of these packages or libraries these aspects must be considered before attempting to integrate it into the system. The consequence for selecting a sub-optimal package which is not suited for the needs of the system could have potential cascading effects and could potentially force the deprecation of some libraries used in the prototype. This ultimately wastes development time and negatively impacts the timeline for completion Many of the architectural and system choices were decided on because they are also used in CARTA's architecture. Which will be explain in the packages respective sections.

4.1 React

A free and open-source JavaScript library is used for building user-oriented applications based on components. It can be employed to develop single-page, mobile, or server-rendered applications. It is maintained by Meta and a community of individual developers as well as companies.

React.js is the chosen front-end framework to implement the web application's front-end for VRDAVis. The motivation behind this decision is to replicate architectural choices from the CARTA system, as CARTA also utilised React.js to implement its front-end application.

4.2 WebRTC

Web Real-Time Communication (WebRTC) Blum et al. (2021) is an internet communication protocol that enables web applications to stream audio, video, or data between peer browser windows, without the need for the data to be routed through an intermediary such as a server. It facilitates peer-to-peer communication without requiring additional third-party packages.

It is used in VRDAVis to establish a connection between a front-end instance on a desktop computer browser and an instance on a standalone headset browser. This connection is then utilised to transmit data regarding the application's state between

the peers, allowing the user to seamlessly continue their work without needing to repeat steps on another device.

4.3 Three.js

The framework Three.js Danchilla (2012) is a cross-browser JavaScript library built on top of WebGL. It is utilised for creating and animating three-dimensional computer graphics within web browsers.

In VRDAVis, it finds application in implementing assets within the virtual environment or three-dimensional space. Three.js is widely adopted and benefits from an active community, offering robust support. Moreover, it can be seamlessly integrated with React.

4.4 WebXR

WebXR provides access to both input data, related to the pose information from the headset and controllers, and output, which is the display within the headset. This technology enables web browsers to facilitate Virtual Reality (VR) and Augmented Reality (AR) experiences over the internet. It encompasses functionalities commonly associated with VR and AR devices.

In VRDAVis, WebXR is employed to integrate an embedded VR environment into the client web application. It works in conjunction with Three.js and React.js to create an immersive three-dimensional environment, constituting the VRDAVis front-end application.

4.5 MobX

This is a state management library designed for front-end web browser applications, responsible for maintaining stateful information. It seamlessly integrates into React and is also utilised for state management within the CARTA framework.

4.6 Node.js

Also refereed to a Node is a free, open-source server environment, with cross-platform capabilities which uses JavaScript to create server-side tools Node uses asynchronous programming which is a different manner to handle HTTP requests compared to PHP or ASP. In Node, the server receives a request task, this request task could be to open a file in the file system. Node send the task to the file system and prepares to handle next request. When the file system has opened and read the file, the server returns the content to the client. Whereas PHP sends the task to the computer's

file system, waits for the file system to complete it's task and returns the content to the client. Only once a task is complete is the server ready to handle the next request. Asynchronous programming eliminates waiting and just continues with the next request.

VRDAVis utilises Node to implement the signalling server to facilitate and manage peer-to-peer connections between the instances of the front-end web applications. It was chosen because simple servers can be implemented quickly. While the tasks which is has to perform are not as performance sensitive as the actions the data server has to undertake, therefore performance and speed are not prioritised.

4.7 Express

Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It is one of the most popular Node web frameworks. Provides functionality to write handlers for HTTP requests at different URL paths also know as routes. It is integrated with view rendering engines to generate responses by inserting data into templates. It has the functionality to set common web application settings like the port for connecting, and the location of templates that are used for rendering the response. It can also add additional request processing know as middleware at any point within the request handling pipeline for additional functionality such as authentication.

It is used on the VRDAVis signalling server to implement the WebSocket connection functionality and was chosen because of its flexibility and the speed at which prototype systems can be implemented.

4.8 C++

A high-level, general-purpose, object-oriented programming language that was initially released in 1985 as an extension of the C programming language, with a focus on performance, efficiency, and flexibility as its primary features.

Its rapid execution and performance-oriented nature are precisely why it is selected for this use-case.

4.9 HDF5 C++ Library

HDF5 is the selected file format for data storage on the remote data server, primarily employed for storing three-dimensional astronomy data cubes. The HDF5 C++ library is utilised to read the files stored on the server. Initially written in C, the

C++ package encapsulates the Python code into a format that can be interfaced with C++.

The decision to use the HDF5 format is based on its emphasis on performance, allowing rapid reading and writing of data to files, as well as the capability to instantly access any part of the file.

4.10 uWebSockets

This is employed in building the C++ web server and enables the establishment of WebSocket connections on the remote data server in C++. This choice was made to align with architectural decisions similar to those of CARTA.

4.11 Protocol Buffer

A language and platform-neutral mechanism for converting data objects into a byte stream for transfer. It enables you to define the structure of messages and facilitates the reading and writing of structured data to and from various data streams using different programming languages.

This technology is used to define a schema and format for data exchanged between the front-end and remote services. It was chosen due to its error-prevention capabilities and greater effectiveness compared to JSON.

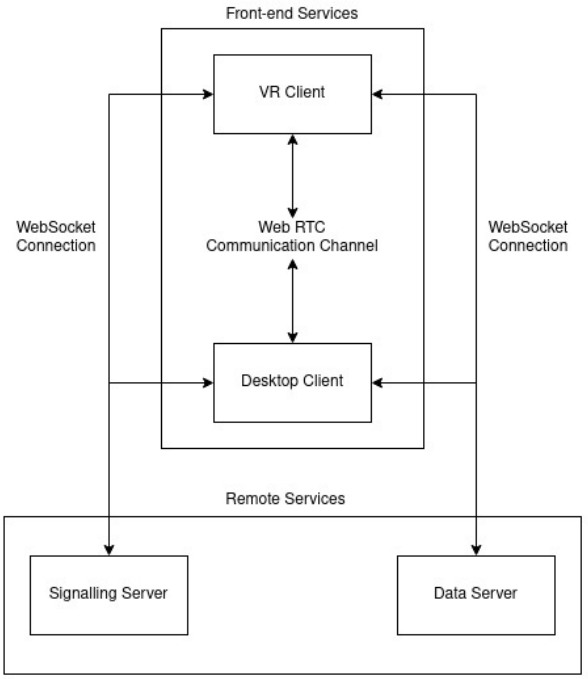

Figure 8: The VRDAVis high-level system architecture, showing the front-end and remote services. The arrows depict communication channels between the components, where the main channels of communication are between the client of the front-end services and the data server of the back-end services, the front-end client and the signalling server, and the VR and Desktop clients.

5 System Design and Implementation

This section explains the design and implementation of the VRDAVis system. It outlines the system's structure and delves into how each component functions, some key language and package choices, as well as some key algorithms. The full source code can be found at the public GitHub repository for VRDAVis van Zyl (2022).

The architecture of CARTA was used as a guide for architectural choices and implementation. VRDAVis is a prototype proof-of-concept system, with the goal of potential integration into CARTA as additional functionality. The architecture was kept as similar as possible to allow the systems to be integrated with minimal changes to both systems.

The system uses the client-server architecture, with two main subsystems, the front-end and remote services. The front-end service only contains the client web browser

application. This application is used on both the desktop computer and the standalone VR headset. These remote services host, deliver, and manage the resources and services requested by the client.

Several libraries were evaluated for the development of VRDAVis. These included libraries for embedding VR environments and rendering three-dimensional graphics in a web browser, web-socket and peer-to-peer connections. Aspects of these packages and libraries such as functionality, documentation, community support; must be considered before attempting to integrate. Selecting a sub-optimal package which is not suited for the needs of the system could have potential cascading effects on the overall design of the system and the project timeline. Many of the architectural and system choices were selected of compatibility with CARTA's architecture, the relevance of each choice will be explained further in the respective sections. Table 1 summarises all the technologies employed in the VRDAVis; what they are, where they are used, what their individual purpose is, and a link to their associated website.

The main decisions in designing and implementing this architecture is deciding which system function are performed by the respective services. Once the system has been implemented, moving a task to a different service could be costly. For this prototype system, only one architecture will be tested.

The remote and front-end services, with their respective components, work together to create VRDAVis. We explain the functionality starting with front-end service, where the user begins their interaction with VRDAVis.

The web browser application is accessible through the Internet, either on a desktop computer or on a standalone VR headset. The interface allows the user to choose from a list of files available on the data server. These files are pre-processed HDF5 files that store data at various levels of resolution. This allows the browser application to request the same section of a data cube at a higher or lower resolution, and then display the requested data for the user to interact with.

The user can use the file selection UI on a desktop computer or a VR headset, but the embedded VR environment can only be accessed through the browser on the VR headset. Once the file has been selected, a low-resolution version of the entire data cube is requested from the data server. The data is sent from the data server to the browser application, where it is rendered into a visualisation. The visualisation can be viewed from the browser window or from within the VR environment.

However, if the user is viewing the visualisation from a desktop computer and would like to view it from a VR headset, the state of the visualisation can be transferred to the VR headset, and vice versa. This process uses a peer-to-peer connection between

the two devices to transfer the application's state. This connection is facilitated by the signalling server, which assists with the pairing of the devices as well as automatically establishing a new peer-to-peer connection. The signalling server itself is not involved with the data transfer once the connection is established.

Once the user has entered the VR environment, they can interact with the visualisation using various actions like translation, rotation, and scaling. If the user wants to examine a portion of the data cube in more detail, they can select the crop mode from the menu and draw a cube around the portion they are interested in. Once the area has been finalised and the crop button has been selected, the application determines which cubelets to request from the data server. As they arrive, they are reassembled into a larger cube and rendered. The user can continue to select and crop smaller portions of the visualised cubes until they reach the highest resolution level available for the selected file.

Name	**Location**	**Purpose**	**Link**
React.js	Client application (front-end services)	Construct the UI of the client application.	`https://react.dev/`
WebRTC	Client application (front-end services), signalling server (front-end services)	Used to establish a peer-to-peer connection between two client applications and send data between them.	`https://webrtc.org/`
Three.js	Client application (front-end services)	Used in the construction of the three-dimensional elements within the VR environment.	`https://threejs.org/`
Node.js	Signalling server (back-end services)	Used to create the HTTP signalling server.	`https://nodejs.org/en`

Continued on next page

Table 1 – continued from previous page

Name	Location	Purpose	Link
Express	Signalling server (back-end services)	A framework used in conjunction with Node.js to streamline the creation of the HTTP API.	https://expressjs.com/
C++	Data server (back-end services)	Used to implement the back-end data server.	https://isocpp.org/
HDF5 C++ Library	Data server (back-end services)	Used for opening and reading HDF5 files from a C++ acpplication.	https://support.hdfgroup.org/documentation/hdf5/latest/_r_m.html
uWebsockets	Data server (back-end services)	Used to implement the WebSocket connections to facilitate data transfer between the data server and client application.	https://github.com/uNetworking/uWebSockets
Protocol Buffer	Communication layer	Used to standardise the messages and serialise the data for transfer between the client application and data server.	https://protobuf.dev/overview/

Continued on next page

Table 1 – continued from previous page

Name	Location	Purpose	Link

Table 1: A summary of the libraries and frameworks used in VRDAVis. The table provides information such as the name of the library or framework, where it is used in VRDAVis either the front-end services, back-end services or communication layer, what its purpose is within VRDAVis, and a link to the affliated website.

5.1 Communication Layer

The sub-systems of VRDAVis need communication channels and protocols to communicate effectively. These channels are established between the components of the front-end service and remote services. The data and signalling servers within the remote services communicate with the client application in the front-end. Peer-to-peer connections are formed between instances of the client within front-end services.

5.1.1 Data Server and Desktop/VR Client Application

The communication between the user-side browser application and the server-side data server takes place through a WebSocket connection, which is more suitable for VRDAVis than the typical request and response or promise and resolve model common in many HTTP-based clients and servers MDN Web Docs (2024). WebSockets allow the respective systems to send messages to each other without requiring a request first to respond. This behaviour allows for many responses to a single request, so the server is not restricted to responding to the client with a single very large message with a large piece of data, it can send smaller pieces of data in individual messages.

Protocol Buffers are used to define the structure of messages and facilitates the reading and writing of structured data to and from various data streams using different programming languages, such as C++, JavaScript, and Python. The Protocol Buffer library was created by Google to serialise structured data and is a platform-neutral mechanism for converting data objects into a byte stream for transfer Google (2024). This library is used to define a schema and format for data exchanged between the front-end and remote services. The Protocol Buffer message definitions are stored in their own folder and are used by the user-side and server-side systems. When a message needs to be sent from the user-side to the server-side or vice versa, the data objects are serialised into a byte stream before they are transferred. Once the

data arrives, it is converted from a byte stream back into a data object. IProtocol Buffers was chosen for its error-prevention capabilities compared to JavaScript Object Notation (JSON), and because it is the chosen communication format used in CARTA.

5.1.2 Signalling Server and Desktop/VR Client Application

The signalling server creates, manages, and establishes peer-to-peer connections between instances of the client browser application on different devices. The communication required between the signalling server and the client application is significantly less complex than the communication between the data server and the client application. The tasks performed by the signalling server are also less time-sensitive than those performed by the data server, and less data needs to be transferred between them. These factors led to the use of the simple request-response communication protocol to be used as well as the use of JSON to format the messages. JSON is a lightweight format for storing and transporting data that is language-independent and easy to understand as it considered "self-describing" JSON (2024).

5.1.3 Client Application and Client Application

Another component of the communication layer is the peer-to-peer connection between two instances of the client web browser application. One of these instances would be in a browser on a desktop computer, and the other would be in the browser on a standalone VR headset.

Web Real-Time Communication (WebRTC) Blum et al. (2021); WebRTC (2024) is an internet communication protocol that enables web applications to stream audio, video, or data between peer browser windows, without the need for the data to be routed through an intermediate server. It facilitates peer-to-peer communication without requiring additional third-party packages. WebRTC is used in VRDAVis to establish a connection between these instances. This connection is used to transmit the application's state between its peers, allowing the user to continue their work seamlessly without needing to repeat steps on another device. This peer-to-peer connection allows the instances to exchange data between instances without having to send it through middleware service such as a server. However, it does require the signalling server to establish and manage the connections.

Headset Pairing The initial pairing process requires both of the VR headset and the desktop computer to be connected to the signalling server. When each devices connects, it sends the server an unique ID, a name, and announces whether it is a VR capable device. The server checks if the device already has a pair and if it does

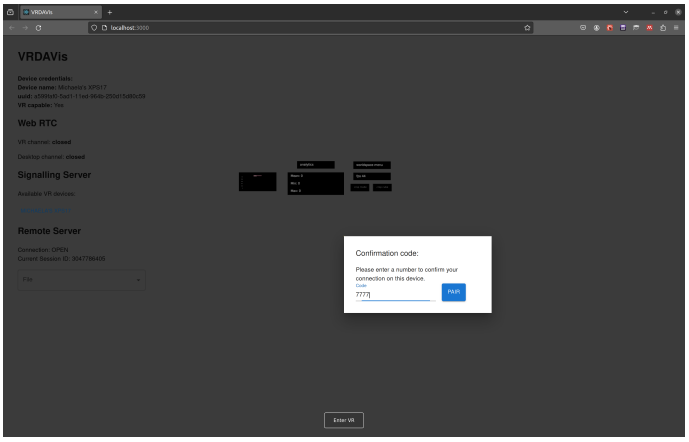

Figure 9: The menu that appears in the browser window when the user selects a VR device to pair with on the desktop computer. The user must enter a code which must also be entered in the same menu on the VR headset.

not find a pair it sends a list of unpaired VR devices to the unpaired devices which are not VR-capable. A desktop user selects a VR device and generates a code from the desktop computer, a code which is sent to the signalling server, and forwarded to selected device using its unique ID. The VR device then prompts the user to enter the code locally, and verifies it against the code received from the server. When the server confirms that the codes match it saves the device pair to a local JSON database. The server then sends a confirmation message to both devices stating that the devices have been paired. Both devices then confirm that they are ready to begin a peer-to-peer connection and the WebRTC peer-to-peer protocol begins.

Paired Devices Connection When one device connects to the signalling server, the server checks if the device is currently paired by searching the JSON file which contains all the pairs. If the device is paired, the server checks if the other paired device is connected to the server as well. When the VR device and the desktop computer are connected to the signalling server and the server detects that these device are paired to each other, it starts the process of creating the WebRTC peer-to-peer connection.

Peer-to-peer Connection Once the VR headset and the desktop client indicate that they are ready to establish a peer-to-peer connection with each other, the desktop client sends WebRTC Interactive Connectivity Establishment (ICE) candidates to the VR device through the signalling server. These ICE candidates contain the protocols and routing needed for remote devices to communicate with each other as well as details used to establish the connection. Multiple candidates are sent

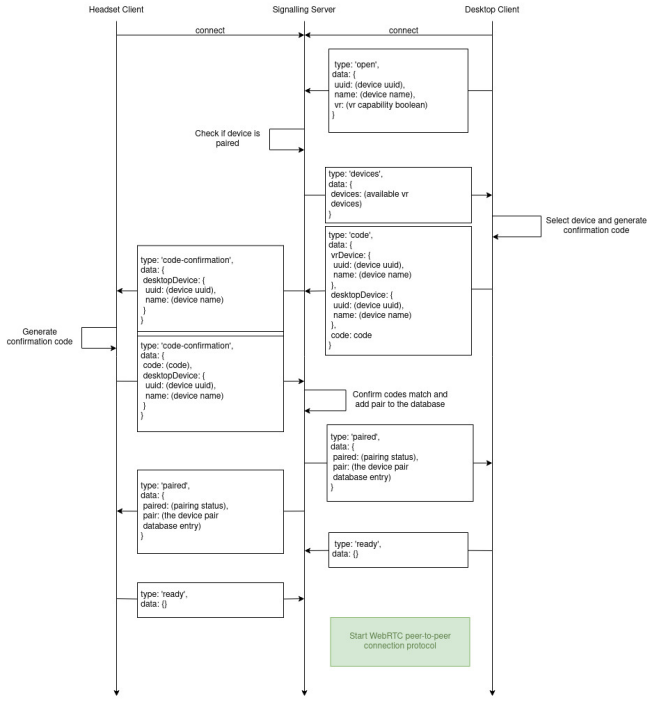

Figure 10: The pairing process between an instance of the front-end application running on a desktop computer and an instance running on a VR headset. It shows the order of the messages sent from each device and the contents of each message.

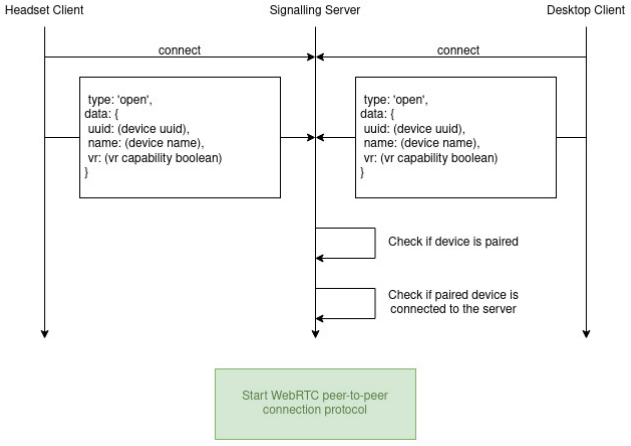

Figure 11: The process of connecting paired devices through the signalling server. Both devices indicate that they wish to open a peer-to-peer connection with their "open" messages.

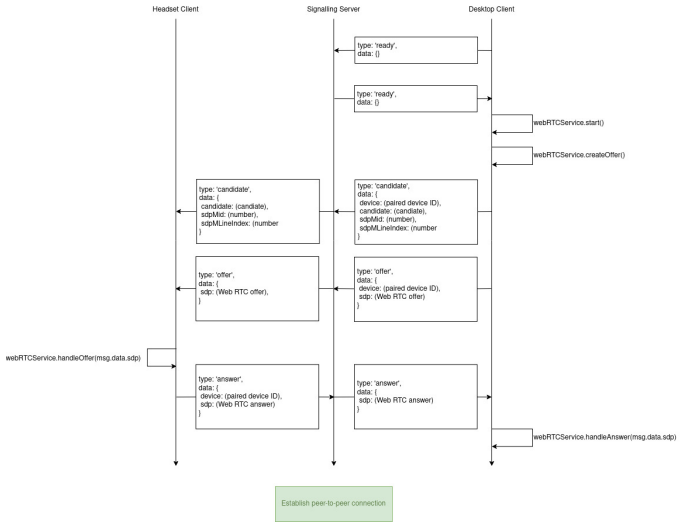

Figure 12: How the peer-to-peer connection is established between the paired devices.

until one is found which works for both devices. The candidates are sent to the VR device through the signalling server as this is the only channel currently connecting the devices. The desktop computer creates a Session Description Protocol (SDP) offer, which contains the RTCPeerConnection interface. It contains information about open channels already attached to the current WebRTC session, a codec, any candidates already gathered, and options supported by the browser. This SDP offer is sent to the VR device through the signalling server, and the VR device handles the offer through WebRTC by creating an answer which contains the corresponding information about its own capabilities. The answer is returned to the desktop client, the offer is resolved and the peer-to-peer connection is established between the devices. No further data needs to pass through the signalling server in order for the devices to communicate with each other; they now pass data to one another through the peer-to-peer connection.

Transferring Client Application State Over Peer Connection The purpose of the peer-to-peer connection is for the connected instances of the front-end to exchange state. This allows the user to start their exploration of a data cube on a device such as a laptop or a desktop computer, then move the state of the desktop instance to the VR headset to continue the exploration in the VR environment. Transferring the state of an instance in this manner means that the user does not need to perform additional steps to start a new instance and get back to the same position in their new exploration.

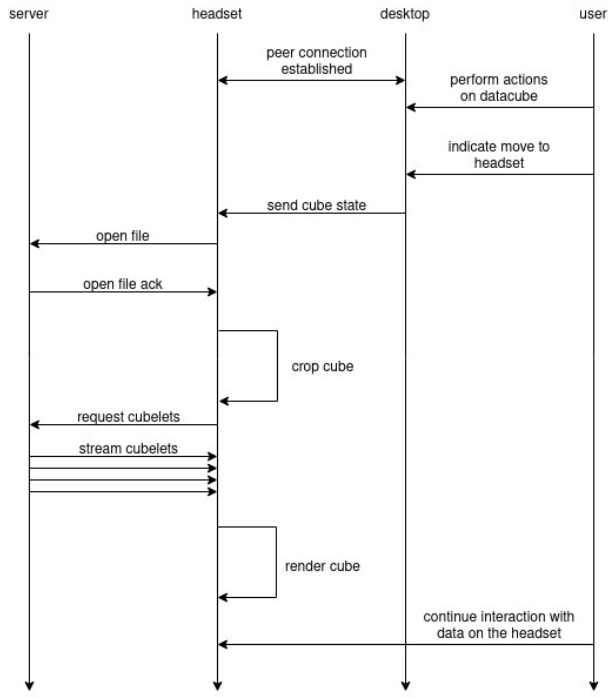

Figure 13: The diagram depicts the event sequence for transferring the state between the two instances of the front-end browser application. The application state can be bilaterally transferred between client devices.

The state is transferred through JSON messages sent over the peer-to-peer connection. Each message contains the file name, and the size and the position of the local cube state. Once these details have reached the new instance, a new connection to the data server is used to make requests and receive responses from.

5.2 Server-side

The remote services of VRDAVis are composed of all the systems that serve the front-end applications; consisting of a data server and a signalling server. The data server receives requests from the user application, where its purpose is to transfer data from astronomy data files. The signalling server is used to create and manage peer-to-peer connections between two instances of the front-end browser application. The peer-to-peer connection is used to transfer the state of the visualisation across instances, allowing the user to transfer the state from a desktop computer to a standalone VR device and vice versa. This showcases how VRDAVis would be able to integrate into an astronomer's workflow with minimal disturbance.

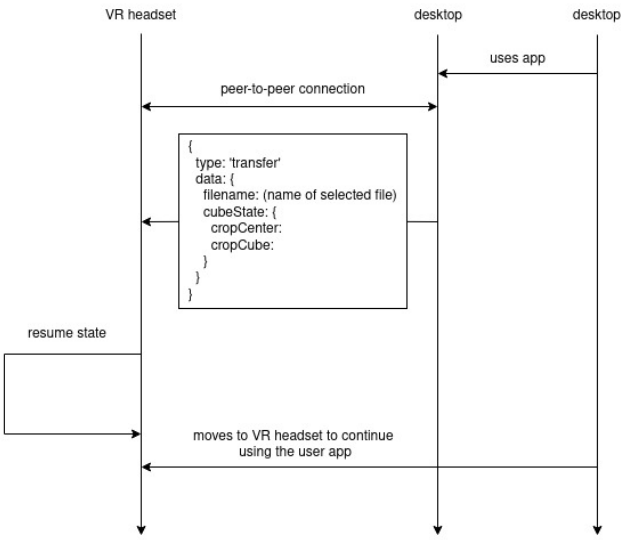

Figure 14: The message structure which is sent between the VR headset instance and the desktop instance when the cube state needs to be transferred between the devices.

5.2.1 Data Server

It is common in astronomy for data to be stored in the FITS format. But in VR-DAVis the HDF5 file format will be used to store data. Unlike FITS files, where data must be read sequentially, the HDF5 format allows any part of the file to be accessed at any point in time. The HDF5 C++ library is used to read the files. Initially written in C, the C++ package encapsulates the Python code into a format that can be interfaced with C++.

However, using a more effective file type does is not speed-up the process of creating a visualisation. Especially when the data in the file is still at its original resolution and therefore its largest size. It has the potential to overwhelm the client device with the sheer amount of data it would need to process. The full-resolution data must be processed into multiple levels of resolution, referred to as mipmaps. These mipmap in the HDF5 file format are generated from FITS files using the*fits2idia* Team (2023) repository provided by the CARTA organisation. They are generated by taking an average of a window of voxels. These mipmaps are contained in the same HDF5 file as the data at its original resolution. Portions of the mipmapped data are read, instead of reading the full-resolution data, decreasing the likelihood of overwhelming a client device.

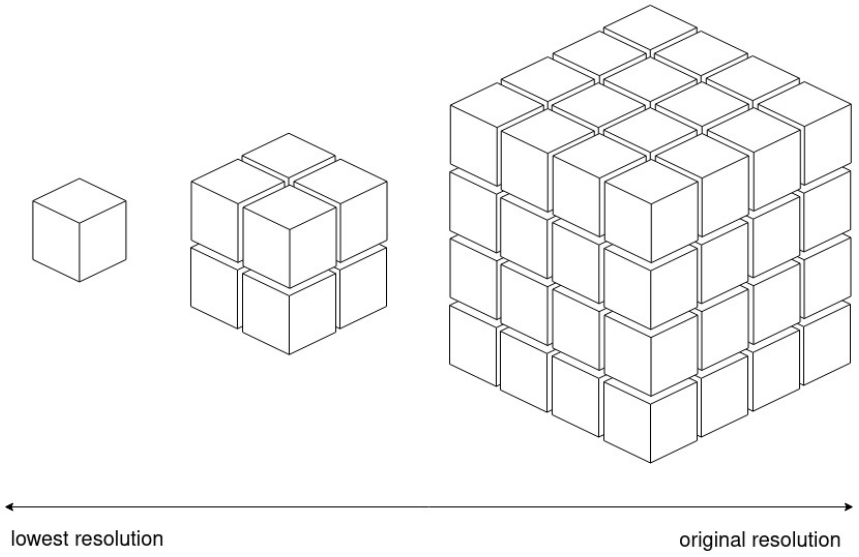

Figure 15: In the pre-processed data file, different resolutions are stored as mipmaps, from a single cubelet which represents the entire data cube to the original full-resolution data, with several intermediate levels.

The server stores the data files pre-processed into a mipmapped format, the mipmaps are only generated once when the data is loaded onto the server and can be used across multiple sessions. This approach to handling *larger than local memory* files allows the entire dataset to be visualised with less computational overhead, compared to rendering the full resolution dataset Li et al. (2016). It still allows the visualisation to maintain the overall context of the data, but as the data is explored, higher resolution versions of the same region can be swapped out to allow the user to observe the details.

The server must fetch data from its filesystem according to the client's request. When the client requests a number of sub-sections of a data cube, the server retrieves these cubes or cubelets and streams them back to the client over a websocket connection. Each message streamed to the client contains one compressed cubelet. Additionally, the server performs file system functions such as sending lists of available files, opening them, and reading data.

The ability to view various analytics on the visualised data is also an important feature, and the server supports this by computing various derived values such as minimum, maximum, and median values. It also compiles analytics data to be visualised as various graphs, including the distribution of values.

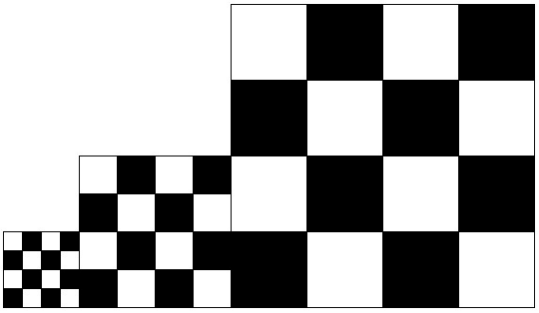

Figure 16: An example of a two-dimensional mipmap. Observe how the pattern is scaled at each level

The data server was written in C++ for the purposes of speed and performance and follows a similar structure to the CARTA software. C++ is a high-level, general-purpose programming language used to implement the data server of VRDAVis remote services. An object-oriented programming language that was initially released in 1985 as an extension of the C programming language, with a focus on performance, efficiency, and flexibility.

It was selected for its speed of execution and object-oriented nature, as well as being the same language utilised in implementing the CARTA back-end. uWebSockets is a C++ library used to integrate WebSockets into a C++ project. It is also used establish WebSocket connections between the client application and the data server.

5.2.2 Signalling Server

Node.js, also referred to as Node, is a free, open-source server environment, with cross-platform capabilities, which uses JavaScript to create server-side tools. It uses asynchronous programming which is a different manner to handle the conventional HTTP one request, one response model. Synchronous systems wait for tasks to complete before returning content to the client, only once a task is complete is the system ready to handle the next request. Asynchronous systems do not wait for a task to be completed before starting a new request.

VRDAVis uses Node in the signalling server to create and manage peer-to-peer connections between the instances of the front-end web applications. It was chosen because servers can be implemented quickly.

Express is a minimal and flexible Node.js framework that provides a robust set of features for the development of web and mobile applications. It is one of the most popular Node web frameworks. It provides functionality to write handlers for

HTTP requests at different URL paths know as routes. It is integrated with view-rendering engines, which generate responses by inserting data into templates. It has the functionality to set common web application settings like the connection port, and the location of templates. It can also insert additional request processing known as middleware at any point within the request handling pipeline, to add functionality such as authentication. It is used in the VRDAVis signalling server to implement the WebSocket connections, and was chosen because of its flexibility and the speed at which prototype systems can be implemented.

5.3 User-side

The front-end service of the system is a user serving web browser application accessible through the internet. The decision to use a browser application ensures that application is operating system-agnostic, which makes it available to more user devices. It also allows the system to remain up-to-date with any changes for all users, unlike native applications, which the user is responsible for updating. The instances of the application can be used on desktop computers, laptops, and standalone VR headsets, and communicate through a peer-to-peer connection, which allows them to coordinate with each other without routing data through a server. These web application instances serve as entry points to the embedded VR environment, which can be accessed through the VR headset. This is the environment where the user can see the visualised data, interact with the visualisation, manipulate it, and explore.

React.js is the framework chosen to implement the web application for VRDAVis, to replicate architectural choices from the CARTA system, as CARTA also uses React.js for its front-end application. It is a free and open-source JavaScript library is used for building user-oriented applications based on components and can be employed to develop single-page, mobile, or server-rendered applications. It is maintained by Meta and a community of individual developers as well as companies.

In addition to React.js, MobX is a state management library designed for front-end web browser applications, responsible for maintaining stateful information. It integrates into React and is also used for state management within the CARTA framework.

The front-end also hosts the majority of the system logic as it determines which piece or pieces of data need to be retrieved from the server. When the initial data is retrieved or the cube is cropped, a list of cubelets must be made so that the correct subsections can be fetched from the appropriate resolution level. The resolution level and lists are determined through a selection process taking into account the predetermined cubelet size, cubelets are selected from a mipmap level which maximises the resolution while maintaining performance. It is biased towards the mipmap layer

with the maximum downsample ratio, which reduces the total number of cubelets that cover a selected area, whether it is the whole cube or a subset of the cube. This strategy limits the maximum amount of data that can be sent to the client.

This selection process is critical when the cube is cropped on the front-end, as it discards portions of the cube that are not being focused on. Figure 17 illustrates the relationship between the cropped portion of the cube and the cube space, showing the dimensions of the full-resolution cube, as well as how multiple cropping actions relate to each other. Each crop action selects a subset of the previous cube.

The algorithm used to crop the cube and select cubelets at a higher resolution level can be broken down into several steps:

1. To determine which cubelets need to be requested initially, the full-sized data cube is cropped with a selection boundary of the same size. The cube localspace is the current state of the cube being visualised on the front-end application, and the crop cube is the subset of the cube for which higher resolution cubelets are going to be requested.

2. Every time the cube is cropped, the visualised cube's space and crop cube's space are aligned in the full-resolution cube's space, which can also be referred to as the cube's worldspace, encapsulating the full resolution dimensions. This is done to ensure that the crop cube's corner coordinates fall within the x, y, and z dimensions of the cube's coordinates. This is achieved through the multiplication of the crop cube dimensions and the centre coordinates by the current mipmap level. The initial mipmap level for the XY and Z axes is set to 1, and when the cube is cropped, the cube localspace dimensions and centre are updated to equal the crop cube's dimension and centre.

3. The corners are calculated by the division of the dimensions by 2 then the addition or subtraction of this number to the centre for the x, y, and z dimensions.

4. The crop cube's dimensions and centre are translated to coordinates within the full-resolution cube.

5. The maximum mipmap level that data can be fetched from is determined by the adjustment of the crop cube corners to coordinates within the data cube, so that the centre of the axis shifts from zero to the midpoint of the full-resolution cube dimensions. This point is referred to as the adjusted centre.

6. The current mipmap for XY and Z planes are initially set to 1. The minimum and maximum values of x, y, and z dimensions are determined to establish which cubelets are fully or partially inside the cropped area.

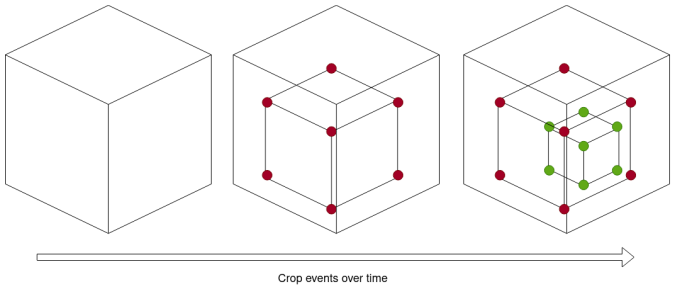

Figure 17: The relationship of the crop cube across cropping events to the full resolution data cube as the user drills down into a section of interest. The cropped sections refer to the marked red and green squares in the diagram; these sections are a selection made during a crop action. They are the portion of the data used to create the visualisation in the front-end services.

7. The crop cube dimensions are rounded to the nearest cubelet boundary.
8. The process then steps through each range from the minimum boundary to the maximum boundary level.
9. The step is divided by the cubelet dimension size to determine the encoded coordinate of the cubelet needed from the data server.

For the user to view the visualisation with the VR headset in a virtual environment, a package is require which makes the three-dimensional environment compatible with VR displays and controllers. WebXR grants access to the display of the headset, as well as the positions of the camera and controllers, allowing for integration with the web. This technology enables web browsers to facilitate Virtual Reality (VR) and Augmented Reality (AR) experiences over the internet. It encompasses functionalities commonly associated with VR and AR devices. Three.js was used to create the assets within the VR environment, such as the data cube used to display the astronomy data, and the UI elements which use canvases as textures to display text on a plane. These elements are made interactive through WebXR functions. The combination of these libraries facilitates a VR experience through the internet. The data visualisation is presented within the VR space as a cube mesh with a three-dimensional texture used to map the three-dimensional data. This presentation method was selected because of its minimal processing cost Ferrand and Warren (2018).

The data visualised by VRDAVis is volumetric astronomy data; however, this system would be able to visualise any volumetric data as long as it is in the HDF5 file format. The HDF5 schema used is the same schema used by the fits2idia repository Team (2023), which is used to convert FITS files into the HDF5 files. The mipmaps in the

HDF5 file are stored according to their downsampling ratios. For example XY_2_Z_4 means that the x and y axis have been downsampled to a size 2 times smaller than the original cube and the z axis has been downsampled 4 times smaller than the original cube. The data arrives at the front-end from the data server in a float array format, where it is placed in a three-dimensional texture and rendered using a ray-cast shading technique. The process of the ray-casting shader has the following steps:

1. A ray is cast until it hits a bounding box of the volume texture.
2. The ray takes a step through the volume and samples a point from the three-dimensional texture.
3. The values that the ray sampled are added to an accumulator to determine the opacity of the pixel.
4. The shader keeps track of the maximum value the ray has sampled.
5. The ray continues to sample points until it has reached its maximum step count or the accumulated opacity reaches a 100%.
6. The shader then samples a two-dimensional colour map texture to retrieve a colour value for the pixel.
7. If the opacity of the pixel is 0%, the point is discarded.

The rendering of the data with a ray-casting shader was implemented with the Three.js framework Danchilla (2012), it can be easily integrated with React.js, a cross-browser JavaScript library built on top of WebGL. It is used to create and animate three-dimensional computer graphics within web browsers. Three.js is widely adopted, and benefits from an active community that offers robust support.

The shader goes through these steps for every frame, for every pixel on the screen. This is the most computationally expensive step in the visualisation process. Although ray-cast shading is computationally expensive, it depicts the data in a way that closely matches its structure, as if it were a high-resolution three-dimensional scatter plot.

In VRDAVis, WebXR is used to integrate an embedded VR environment into the client web application. It works in conjunction with Three.js and React.js to create an immersive three-dimensional environment, constituting the VRDAVis front-end application.

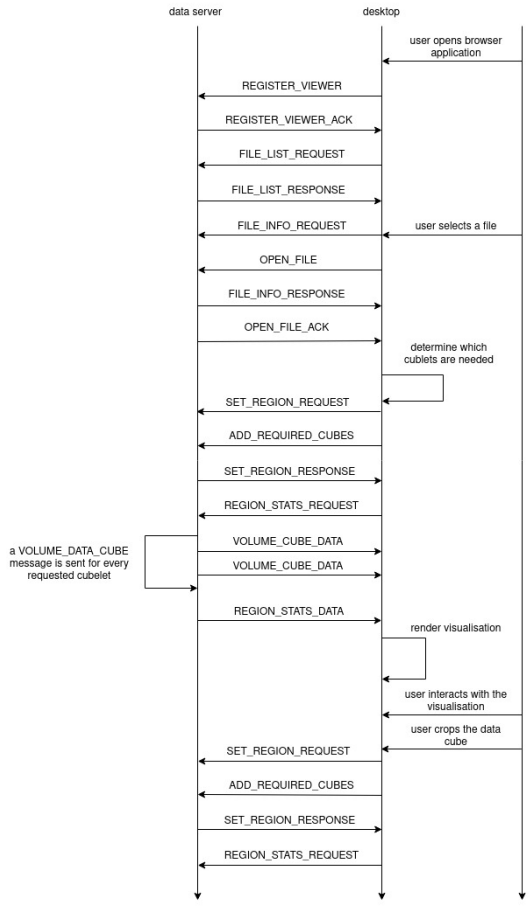

Figure 18: A sequence diagram showing the user's interaction with an instance of the client application and the corresponding protocol buffer messages which are sent from the front-end to the back-end, as well as the responses from the back-end to the front-end. It shows the message sequence of the user opening the front-end application, selecting a file from a list, and cropping the visualisation.

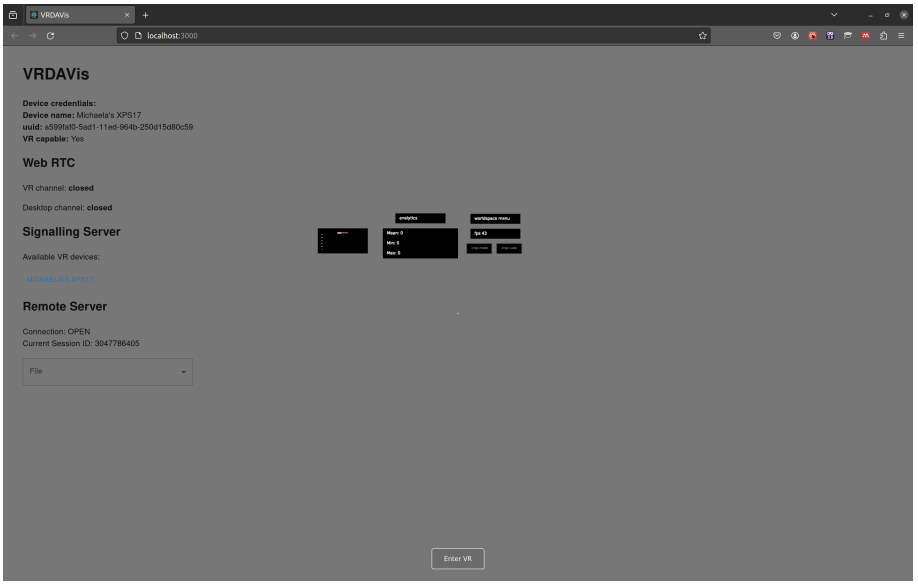

Figure 19: The appearance of the VRDAVis application when it is fist opened in the browser. In this instance the browser is Firefox. Some information about the current session can be seen on the left side of the screen, as well as the status of the peer-to-peer connection under the Web RTC heading, the available VR devices which can be paired with under the Signalling Server heading, and a drop down menu for selecting a data cubes under the Remote Server heading. Some VR environment assets can be seen in the centre of the screen.

5.3.1 User Workflow

The user first opens an instance of the browser application on their desktop computer. When it is launched, this application instance creates a connection with the remote services such as the data and signalling servers. The signalling server connects the desktop client instance and pairs with the standalone VR headset if it is also connected to the signalling server at that time. This connection is made automatically once both devices are connected to the server. If the desktop device is not paired, then the user can pair it with an available unpaired device. When this peer-to-peer connection is created, the state of the visualisation can be transferred between the paired devices. The data server acknowledges the client application's connection, and the client requests a list of the available files on the server.

The data server sends the file list, and the user can choose which file they would like to explore. The user's file choice is sent to the data server, and the data server sends back some information pertaining to the selected file, such as dimensions and the size of the file. The client determines which cubelets it requires to generate the

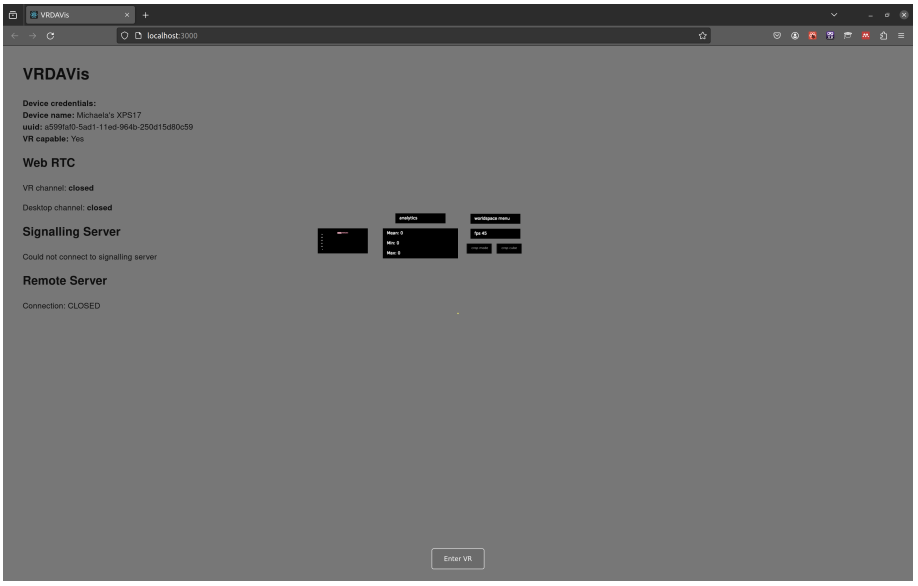

Figure 20: The appearance of the browser application when none of the remote services are available; there is no peer-to-peer connection, no connection to the signalling server, and no connection to the remote server.

initial visualisation and requests them from the data server, while also specifying which analytics the data server needs to compute using the full-resolution data. The data server opens the file, reads the cubelets the client requested, compresses these cubelets into individual messages, and sends them to the client. Once the analytics are computed, they are also sent to the client.

When a cubelet reaches the client, it is decompressed and loaded into the GPU cache. The cubelet is then added to the three-dimensional texture, one at a time, until all the cubelets have been received and added. This texture is then rendered by the ray-casting shader.

The user can then choose to switch to viewing the visualisation on the standalone VR headset. If the user switches to the headset, all the state information is sent over the peer-to-peer connection.

The headset client uses the state information to start its own session on the data server and request the same cubelets as on the desktop client instance. The VR instance does not need the desktop instance to function; the user can choose to start the entire visualisation process on the VR headset by starting the client browser application on the browser. The main difference between the instance of the application on the desktop and the VR headset is that the VR headset client instance has the entry point to the VR environment, whereas the desktop does not have the

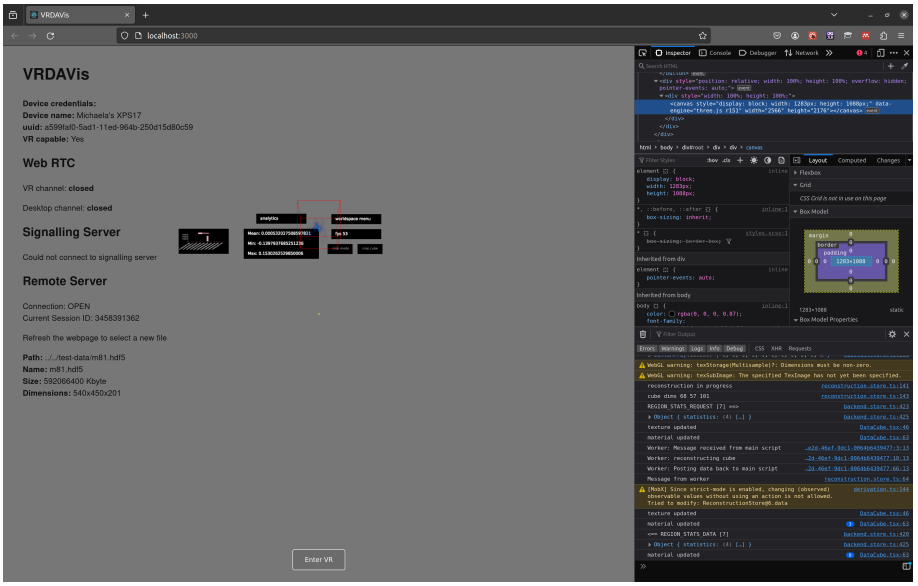

Figure 21: The view of the web browser application when a file has been selected and an initial visualisation of the data cube is within the VR environment. Some additional information about the data cube is displayed under the Remote Server heading; the path of the file on the remote server, the name of the file, the size in kilobytes, and the dimensions of the data cube.

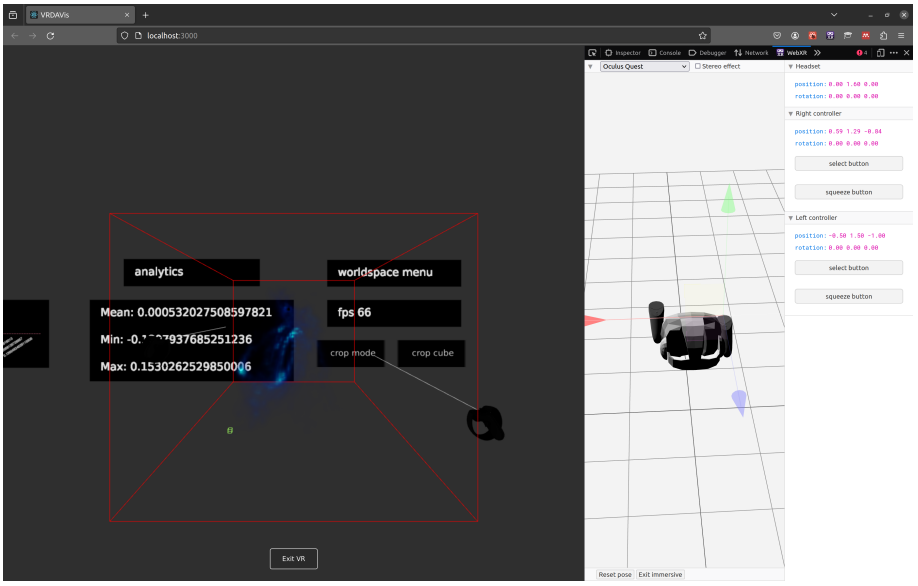

Figure 22: The view from inside the VR environment. The assets which could be seen on the centre of the view are now tangible within the VR environment. The data-cube is now a three-dimensional object and the analytics menu can be seen behind it. The WebXR browser plugin is used to view the VR environment from the browser window on a conventional screen. The plugin interface is displayed on the right side of the browser window.

capabilities to view a VR environment.

When the user enters the VR environment, they can interact with the visualisation in a three-dimensional space and view the analytics of the data. The user can crop the visualisation to get a higher-resolution version of the data by using the crop function; the user drags a box over the section they would like to take a closer look at and then selects the crop button. The client application determines which resolution level is appropriate for cubelet selection and sends a list of cubelets to the data server. The server fetches the cubelets and computes the analytics in the same way as it did for the initial visualisation. This process happens every time the cube is cropped until it reaches the highest resolution level, which is the full-resolution data. The size of the cubelets is determined by predetermined, internal values in the remote and front-end services.

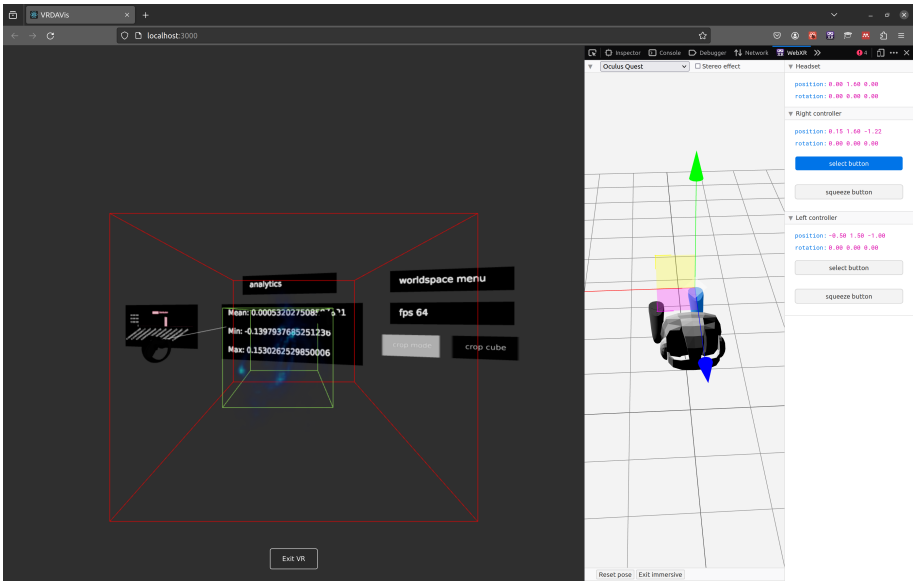

Figure 23: An example of a data cube displayed in the VR environment. The red cube represents an area within full dimensions of the data cube, and the green cube shows the selected crop area.

6 Initial Evaluation

The evaluation was designed to judge the sentiment of expert users and open a line of communication to receive feedback and guidance on future improvements. These expert users were individuals who are researchers in the astronomy field; they also required some experience with VR and experience of using a similar system such as iDaVIE. They are aware of the requirements of a successful VR experience, as well as a successful astronomy data visualiser. VRDAVis is an experimental system and has some shortcomings that may confuse a novice user. The VRDAVis system is optimised for user experience on the standalone VR headset, which makes their feedback crucial. Semi-structured interviews guided the participants through their interactions with VRDAVis, allowing them to progress at their own pace. Additionally, some questions were asked once they had finished their evaluations to further refine the system based on their insights and experiences.

n the experiment, the participant was asked to start using the system on a laptop and then move to the VR headset. The participant was required to complete tasks such as selecting a file to view on the laptop, transferring the state to the VR headset, entering the VR environment via the VR headset, and manipulating the visualisation of the file. This manipulation involved translating, rotating, and scaling the visualisation, as well as performing an action where the visualisation's resolution

was scaled up as it was cropped. All participants used the same system to ensure consistency in the experiment.

6.1 Participants

The seven participants in the experiment were pre-selected from members of the Institute of Data-Intensive Astronomy. All participants were briefed on the tasks they would perform as well as the protocol to follow if they start feeling nauseous while using the VR headset. Prior to the user evaluations an impact assessment as well as a draft of the consent form was submitted to the University of Cape Town' Research Ethics Committee and ethical clearance was given for the tests to proceed. All participants were asked to sign the approved consent form before the commencement of the evaluation. It reiterates the tasks they will performing and what data will be collected during the experiment, including audio and video recordings.

6.2 System Set-up Description

To conduct the evaluation, a carefully designed set-up is essential to ensure smooth transitions between the laptop and VR headset, as well as to capture the participants' interactions and feedback accurately. The environment is a quiet, controlled space free from external disturbances, equipped with chairs and adequate lighting. The equipment includes a laptop with the necessary software for viewing and manipulating the selected file, along with a standalone VR headset and the required controllers. Participants will begin by selecting and viewing the file on the laptop, then transition to the VR headset and interact with the visualisation. These actions include translating, rotating, scaling, and cropping the visualisation. The procedure involves a briefing area where participants are informed about tasks and protocols, including handling simulator sickness, and sign consent forms. The task area houses the laptop, while a designated transition spot allows for easy movement to the spacious VR area. Monitoring and support are provided to the participants by the conductor of the experiment, who observes for signs of discomfort and offer immediate assistance, with video and audio recording equipment capturing interactions and feedback. Post-evaluation, participants relax in the briefing area where they discuss their experience and answer some questions from the conductor. Clear guidelines are in place for handling simulator sickness, including the option to stop and resume the evaluation. This set-up ensures a controlled, supportive environment, prioritising participant well-being and enabling the collection of detailed feedback on the VRDAVis system.

6.3 Metrics

During the evaluation, several metrics were collected to assess the VRDAVis system. Qualitative data about the user's experience was gathered, including any highlights or noteworthy comments they made while using the system. Observations were made to identify which tasks participants struggled with the most and how well they interacted with the user interface. Feedback was collected on their impressions of seeing the data cube visualised in three-dimensional space and their thoughts on interacting with it. Participants were asked for their opinions on the overall workflow of the system, providing valuable insights into its usability and effectiveness.

7 Evaluation Results

The goal of the evaluation was to gain insight and feedback from users regarding the VRDAVis system. Feedback was gathered by having users evaluate the VRDAVis system, and the results highlight both the successful parts and areas for improvement.

Participant 1 noted a positive experience with the visual quality and interactive aspects of VRDAVis, specifically praising the ease of use for cropping and analysing different data segments. They highlighted the significance of having an immersive experience for visualising complex data, which they found to be more intuitive compared to traditional two-dimensional methods. However, they mentioned the importance of improving the system's stability, as they experienced some technical glitches. They also suggested that additional features, such as more sophisticated data manipulation tools and enhanced resolution, could further enhance the usability and effectiveness of VRDAVis.

Participant 2 found the initial experience counter-intuitive, primarily because they were accustomed to the control mapping of iDaVIE. Despite the learning curve, they did not experience motion sickness and appreciated the smoothness of the system on the VR headset. They highlighted the need for better visual definition and clearer data details. The participant emphasised the importance of being able to reduce data before immersion in VR and noted that VRDAVis and iDaVIE have complementary strengths, suggesting a potential marriage of the two systems for handling big data effectively.

Participant 3 did not encounter significant struggles with the headset, noting that familiarity would improve with use. They suggested that the system could benefit from enhanced contrast to improve the readability of the visualisation. Comparing VRDAVis to iDaVIE, they found iDaVIE to be more feature-rich but appreciated VRDAVis' ease of accessibility.

Participant 4 was impressed by the system but faced difficulties in rotating the data and managing the interface when the cube was ocasionally obstructed by the menu. They found the workflow from laptop to VR headset to be advantageous, particularly for its ability to provide a more immersive visualisation. They noted that while VRDAVis still had some user interface issues, it had the potential to be a helpful tool for quick looks at data. Comparing VRDAVis to iDaVIE, they found iDaVIE to be more streamlined in terms of manoeuvring and visual clarity, which is likely due to iDaVIE using a specialised standalone computer to run on.

Participant 5's main struggle was initially handling the zoom and moving the menu but found these manageable over time. They appreciated the quality of the image

used for visualisation. They noted that VRDAVis' interface was simpler to access compared to iDaVIE, which they found cumbersome to open and load data. They valued the integrated workflow between the laptop and VR headset, highlighting its convenience for quickly checking data without needing an elaborate setup.

Participant 6 encountered an issue with the crop mode activation at the beginning of the evaluation but found the problem minor. They appreciated the efficient workflow that allowed quick transitions between the laptop and VR headset, contrasting it favourably with iDaVIE's longer loading times. They found VRDAVis similar to iDaVIE in terms of basic functionality, though they acknowledged iDaVIE's additional features like moment maps. They were satisfied with the quality of the visualisation, noting its ability to display different intensities and emissions clearly.

Participant 7 noted the placement of some buttons as a minor issue since they sometimes interfered with the data cube. The primary issue was the crop button being obstructed by the cube, but the participant found this manageable by moving the cube. They found the workflow seamless once the cube was uploaded, appreciating the efficiency. Having used iDaVIE before, they saw the potential benefits of VRDAVis in eliminating intermediary steps for data handling. They were satisfied with the visualisation quality, though they noted that scaling the cube was not always smooth.

7.1 Key findings

User Interface and Interaction Some users had some initial difficulty with controls, such as zooming, moving menus, and the unstable crop function. Interface elements sometimes obstructed the view of the data cube and vice versa. Some suggested adjustments made by the users to make the user interface more intuitive and less obstructive, possibly repositioning or redesigning the interface elements to enhance usability.

Workflow Efficiency The ability to transfer the workspace state from the laptop or desktop computer and to the VR headset was seen as quick and efficient by the users. They also appreciated the seamless switch between the respective environments. When the workflow was compared to iDaVIE's, it was noted that iDaVIE requires more time to load data cubes and has more complex steps. VRDAVis was praised for its simpler and faster setup, which can be beneficial for quick analyses.

Visualisation Quality Generally, the quality of visualisation in VRDAVis was considered good. The participants were able to discern different intensities and

areas within the data cube. It was found that while the visual quality was good, iDaVIE's specialised hardware provides better resolution.

System Capabilities and Features The current features of VRDAVis, such as cropping and visualising different data sections, were appreciated. The system was seen as basic yet capable of handling regular astronomy tasks efficiently. The users emphasised the need for more features for adjusting visual contrast, manipulating data cubes, and additional analytical tools similar to ones implemented in iDaVIE.

General Sentiments and Recommendations Participants found VRDAVis to be a promising tool, particularly for its speed and ease of use. The system's ability to offer quick insights without extensive setup was also highly valued, and the prospect of greater independence and flexibility in data handling was considered a significant advantage. The users also provided constructive feedback which suggested user interface adjustments, enhancements in visual quality and smoother interactions, more features, and improvements to the overall user experience.

In summary, while VRDAVis is seen as a user-friendly and efficient tool for data visualisation in VR, improvements in user interface design, available feature, and interaction smoothness can enhance its effectiveness and user satisfaction. The participants appreciated the system's potential to streamline workflows and provide quick, clear three-dimensional visualisations of astronomy data.

8 Conclusion

The first research question asks if the proposed architecture is appropriate for the goals of the system, which is to view three-dimensional visualisations of radio astronomy data and to interact with the visualisation within a VR environment.

VRDAVis implements a client-side rendering architecture which sends data from a remote server to a user application. The data is then rendered in the user application on a client device, which is a low to mid-range laptop or an Oculus Quest 2 VR headset. The visualisation rendered on the client is dependent on the processing power of of the client device, and these devices are comparatively lower-powered in comparison to the dedicated systems which render full-resolution data cubes. This resource constraint limits the amount of data that the client device can receive and process before its performance is negatively affected. Unsatisfactory client performance puts the user at risk of simulator sickness, which is unacceptable. The amount of data that is sent to the client device is therefore limited. This limited amount is adjusted to fit the computational capabilities of most client devices. This ensures that the client will only have to process a fixed amount of data at any given time. The same amount of data will be sent to the client at any given point in time regardless of the size of the full resolution data cube stored in the remote server.

The data requested by the client system is stored as a pre-processed file in a remote server. Mipmaps are used to store data at various resolution levels, from full resolution broken up into fixed sized cubelets to the full data cube compressed into a single cubelet. This means that the client system can request cubelets at an appropriate level which will not overwhelm the client device with too much data. This pre-processing of large data files is vital to the operation of the VRDAVis system, as this allows the client to retrieve the data that is needed at that point in time.

Compressing the data before it is sent to the client device reduces the amount of data transferred over the internet. The time the client is waiting for data is then dependent on the bandwidth of the network connection, reducing the amount of data sent over the connection reduces the amount of time it takes to reach the client. The caching system, although underutilised, allows the system to load data from the cache instead of making a request to the data server when that cubelet is needed. Making a request to the server and waiting for the data to arrive requires much more time than fetching the locally stored cubelet from cache. The caching system also stops the web application from fetching the same piece of data more than once.

None of the users noted that their workflow was interrupted by the need to wait for for the data to be transferred from the server and then processed into a visualisation on the VR headset. Therefore, this system architecture has shown that it implements

a capable strategy for exploring very large data cubes.

The second question asks if the system can generate the visualisations and handle user interaction without the assistance of specialised hardware such as a co-located computer.

The system's performance remained steady across cube of varying sizes. The interaction time perceived by the participants also remained consistent regardless of the size of the cube being accessed on the remote server. The participants did not highlight any major performance issues while viewing the data cube within the VR environment. They also noted that there was no frame-rate lag while they were using VRDAVis. Although the visualisation which VRDAVis produced was not as high-quality as the visualisation iDaVIE produced, they still found it satisfactory for viewing the internal structures of the data-cube and interacting with the visualisation. The feedback from the participants noted satisfactory visualisation and interaction quality on the standalone headset. Therefore, the VRDAVis system can produce quality visualisations and handle user interactions without the need for a co-located computer to perform processing.

The final question asks how VRDAVis compares to a system with similar functionality. The visualisation system iDaVIE was selected for this comparison, all of the evaluation participants had experience using the iDaVIE system.

The main comparison mentioned by participants in the evaluation were the user interface, controls, visualisation quality, workflow, system maturity, and usability.

Participants found iDaVIE's controls more intuitive and streamlined compared to VRDAVis. They mentioned that the action of grabbing and interacting with data in iDaVIE felt more natural. Participants noted some initial difficulties with the controls in VRDAVis, such as zooming, rotating, and moving the menu. However, they also acknowledged that some of these issues come from the learning curve of becoming acquainted with a new system and could be overcome with some practice.

The visual clarity and resolution of data in iDaVIE was considered to be of better quality than in VRDAVis. This was attributed to iDaVIE running on a higher-powered computer, which allowed for more detailed and clearer visualisations. While the visualisation quality in VRDAVis was considered good, it was noted that the system did not have sufficient contrast for the details to be seen clearly.

Loading large data cubes into iDaVIE was described by the participants as time-consuming. However, once the data was loaded, the system's capabilities and features were appreciated. Participants appreciated the workflow presented by VRDAVis, which allowed for seamless transitions between the desktop computer and the VR headset. They found this ability to move quickly between devices to be a

significant advantage, which made it easier to visualise data in VR quickly without the need for prolonged loading times.

iDaVIE was considered to be the more mature and feature-rich system with advanced data manipulation tools and analytics. VRDAVis was recognised as an experimental system, still in need of further development. Participants acknowledged its potential and the advantages of VR visualisation but noted that it lacked some of the advanced features and polish of iDaVIE.

Participants felt more comfortable using iDaVIE because of familiarity and a more refined user interface. The learning curve was described as less steep for those already accustomed to the system. Although some participants experienced a learning curve, they found that VRDAVis became more manageable with use and enjoyed the seamless seamless workflow with the advantages of VR for immersive data exploration. Future improvements in interface design, visualisation quality, and feature expansion could mature VRDAVis into a system similar to iDaVIE.

In conclusion, VRDAVis fulfils its goals as a system for exploring large astronomy data cubes, as it implements an architecture which maintains a standard of performance across various large data cubes on lower-powered hardware such as laptops and VR headsets, particularly the Oculus Quest 2. While user testing confirmed its functional similarity to a mature system such as iDaVIE, it also acknowledged that VRDAVis is experimental and not yet ready to be used in a research context. The VRDAVis system requires more development to add more features and improve the existing ones.

8.1 Future Work

Because of the time constraints imposed on the development of the prototype system, decisions about the system architecture had to be made prior to implementation. Some of these decisions might not have been optimal in hindsight, but were chosen as they still fulfilled the project's objectives. Some of the problems encountered during the development process could be addressed through more development iterations.

User Interface Improvements It was found that the VRDAVis system would benefit from some general polish to the user interface in the VR environment. The user interface elements would sometimes interfere with some of the user's actions, causing some frustration. Participants also mentioned difficulties with initial handling, such as zooming, rotating the data, and moving the menu. Making these controls more intuitive and user-friendly would improve the overall experience. Also, ensuring that buttons and controls are not obstructed by the cube or other elements would enhance usability. This includes repositioning or redesigning interface

elements to prevent accidental activation.

Even though the user can see the larger context of the dataset during the initial visualisation, the context can still be lost as the user drills further into the dataset. A mechanism for keeping track of where the user is within the larger context of the data cube could help alleviate confusion from the user's perspective.

Visualisation Enhancements A common request was for the contrast and clarity of the visualised data to be improved, especially for high-density and low-density features. This involves the implementation of a more advanced and interactive colour transfer function to manipulate the appearance of the visualisation.

Feature Expansion The addition of more features could be added for data manipulation, such as advanced cropping options, moment maps, and the ability to adjust visualisation parameters within the VR environment. As well as a feature providing options to view both masked and unmasked data.

Workflow Optimisation Participants appreciated the ability to transfer state between devices. However, the state transfer is only supported in one direction, and the user must transfer the state manually using interface commands. To improve on this feature must be expanded to include more variables, support bidirectional transfer, and sync automatically when the state changes, without the need for user intervention. The efficiency data loading can also be improved, if users are provided an option to upload their own data. This would reduce the reliance on middlemen to upload and process the data for the users. Participants noted that iDaVIE sometimes takes a long time to load data, so optimising this process for VRDAVis would be beneficial.

Stability and Performance Some technical issues around system stability need to be addressed to ensure that the system runs smoothly without the need for constant reloading. An area that requires notable improvement is the cropping functionality, the main problem is that the algorithm used to selected the required cubelets is not robust enough.

The current implementation of this algorithm is somewhat rudimentary and does not account for some corner cases. It attempts to select cubelets which do not exist in the bounds of the preprocessed data cube and in some cases the cubelets fetched does not match the space the user selected in larger cube. Also, considering the limitations of VR hardware compared to higher-powered computers, future iterations must find ways to optimise the system to run more efficiently on VR headsets.

The cache system implemented in the front-end services was not effectively utilised. This is because the system does not allow the user to zoom out of the visualised data-cube, only to zoom in. This means that the cache system continuously fetched new data and did not utilise data already in cache. The cache system was also reset often because the system was at times unstable and required many refreshes to reset the web application.

Iterative Development Integration with a production system will need to address the issues uncovered during the evaluation. Continuous development with continuous integration with more user testing and feedback would incrementally improve upon shortcomings. The continuous incorporation of user feedback must be included into the development cycle to refine and improve the system. Placing users at the centre of the development process will ensure that the system is considered usable. This type of development cycle is much more in line with how production software and systems are currently made.

References

F. Abidi, N. Polys, S. Rajamohan, L. Arsenault, and A. Mohammed. 2017. Remote High Performance Visualization of Big Data for Immersive Science. *High Performance Computing Symposium (HPC 2018).* https://doi.org/10.22360/SpringSim.2018.HPC.006

J. Ahrens, B. Geveci, and C. Law. 2005. *ParaView: An End-User Tool for Large-Data Visualization.* Elsevier, 717–731. https://doi.org/10.1016/B978-012387582-2/50038-1

A. Ali, J. Qadir, R. ur Rasool, A. Sathiaseelan, A. Zwitter, and J. Crowcroft. 2016. Big data for development: applications and techniques. *Big Data Analytics* 1 (12 2016), 1–24. Issue 1. https://doi.org/10.1186/S41044-016-0002-4/TABLES/2

E. Baracaglia and F. P. A. Vogt. 2019. E0102-VR: exploring the scientific potential of Virtual Reality for observational astrophysics. (11 2019). http://arxiv.org/abs/1911.04500

R. A. Becker, W. S. Cleveland, and A. R. Wilks. 1987. Dynamic graphics for data analysis. *Statist. Sci.* 2 (1987), 355–383. Issue 4. https://doi.org/10.1214/SS/1177013104

N. Bikakis. 2018. Big Data Visualization Tools. (1 2018). https://doi.org/10.1007/978-3-319-63962-8_109-2

N. Blum, S. Lachapelle, and H. Alvestrand. 2021. WebRTC - Realtime Communication for the Open Web Platform. *Queue* 19 (2 2021), 77–93. Issue 1. https://doi.org/10.1145/3454122.3457587

D. Bowman and R. McMahan. 2007. Virtual Reality: How Much Immersion Is Enough? *Computer* 40 (08 2007), 36 – 43. https://doi.org/10.1109/MC.2007.257

F.P. Brooks. 1999. What's real about virtual reality? *IEEE Computer Graphics and Applications* 19 (1999), 16–27. Issue 6. https://doi.org/10.1109/38.799723

E. G. Caldarola and A. M. Rinaldi. 2017. Big data visualization tools: A survey: The new paradigms, methodologies and tools for large data sets visualization. *DATA 2017 - Proceedings of the 6th International Conference on Data Science, Technology and Applications* (2017), 296–305. https://doi.org/10.5220/0006484102960305

A. Comrie, K. Wang, S. Hsu, A. Moraghan, P. Harris, Q. Pang, A. Pińska, C. Chiang, T. Chang, Y. Hwang, H. Jan, M. Lin, and R. Simmonds. 2021. CARTA: The Cube Analysis and Rendering Tool for Astronomy. (6 2021). https://doi.org/10.5281/ZENODO.4905459

Brian Danchilla. 2012. *Three.js Framework*. Apress, 173–203. https://doi.org/10.1007/978-1-4302-3997-0_7

D. Dumitrescu and c. Boiangiu. 2019. A Study of Image Upsampling and Downsampling Filters. *Computers* 8 (4 2019), 30. Issue 2. https://doi.org/10.3390/computers8020030

T. Ertl, R. Westermann, and R. Grosso. 1999. Multiresolution and hierarchical methods for the visualization of volume data. *Future Generation Computer Systems* 15, 1 (1999), 31–42. https://doi.org/10.1016/S0167-739X(98)00046-6

W. M. Farr, P. Hut, J. Ames, and A. Johnson. 2009. An Experiment in Using Virtual Worlds for Scientific Visualization of Self-Gravitating Systems. (5 2009), 49–58. http://arxiv.org/abs/0905.1066

A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J. C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, J. Buatti, S. Aylward, J. V. Miller, S. Pieper, and R. Kikinis. 2012. 3D Slicer as an image computing platform for the Quantitative Imaging Network. *Magnetic Resonance Imaging* 30 (11 2012), 1323–1341. Issue 9. https://doi.org/10.1016/J.MRI.2012.05.001

G. Ferrand, J. English, and P. Irani. 2016. 3D visualization of astronomy data cubes using immersive displays. (7 2016). http://arxiv.org/abs/1607.08874

G. Ferrand and D. Warren. 2018. Engaging the Public with Supernova and Supernova Remnant Research Using Virtual Reality. (11 2018). http://arxiv.org/abs/1811.01542

D. Fisher, I. Popov, S. M. Drucker, and M. C. Schraefel. 2012. Trust me, i'm partially right: Incremental visualization lets analysts explore large datasets faster. *Conference on Human Factors in Computing Systems - Proceedings* (2012), 1673–1682. https://doi.org/10.1145/2207676.2208294

M. Glueck, A. Khan, and D. J. Wigdor. 2014. Dive in! *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 561–570. https://doi.org/10.1145/2556288.2557195

R. Gooch. 1996. Karma: a Visualization Test-Bed. In *Astronomical Data Analysis Software and Systems V (Astronomical Society of the Pacific Conference Series, Vol. 101)*, George H. Jacoby and Jeannette Barnes (Eds.). 80.

R. Gooch. 2006. Karma Users Manual. *Australia Telescope National Facility* (2006).

A.A. Goodman. 2012. Principles of high-dimensional data visualization in astronomy. *Astronomische Nachrichten* 333 (6 2012), 505–514. Issue 5-6. https://doi.org/10.1002/asna.201211705

Google. 2024. Protocol Buffers. https://developers.google.com/protocol-buffers Accessed: 2024-05-20.

A. Hassan and C. J. Fluke. 2011. Scientific Visualization in Astronomy: Towards the Petascale Astronomy Era. *Publications of the Astronomical Society of Australia* 28 (1 2011), 150–170. Issue 2. https://doi.org/10.1071/AS10031

A. H. Hassan, C. J. Fluke, D. G. Barnes, and V. A. Kilborn. 2013. Tera-scale astronomical data analysis and visualization. 429, 3 (March 2013), 2442–2455. https://doi.org/10.1093/mnras/sts513 arXiv:1211.4896 [astro-ph.IM]

M. L Heilig. 1994. United States Patent office: stereoscopic-television apparatus for individual use. *ACM SIGGRAPH Computer Graphics* 28, 2 (1994), 131–134.

T. F. Iskandar, M. Lubis, T. F. Kusumasari, and A. R. Lubis. 2020. Comparison between client-side and server-side rendering in the web development. *IOP Conference Series: Materials Science and Engineering* 801 (5 2020), 012136. Issue 1. https://doi.org/10.1088/1757-899X/801/1/012136

T.H. Jarrett, A. Comrie, L. Marchetti, A. Sivitilli, S. Macfarlane, F. Vitello, U. Becciani, A.R. Taylor, J.M. van der Hulst, P. Serra, N. Katz, and M.E. Cluver. 2021. Exploring and interrogating astrophysical data in virtual reality. *Astronomy and Computing* 37 (2021), 100502. https://doi.org/10.1016/j.ascom.2021.100502

W. Joye, oldherl, D. Burke, and K. Glotfelty. 2025. *SAOImageDS9/SAOImageDS9: v8.7b1.* https://doi.org/10.5281/zenodo.14611298

JSON. 2024. Introducing JSON. https://www.json.org/json-en.html Accessed: 2024-05-20.

A. E. Kaufman. 1996. Volume visualization. *ACM Computing Surveys (CSUR)* 28, 1 (1996), 165–167.

A. E. Kaufman. 2000. *Volume Visualization in Medicine.* Elsevier. 713–730 pages. https://doi.org/10.1016/B978-012077790-7/50050-3

D. A. Keim. 1997. Visual techniques for exploring databases. In *Knowledge Discovery in Databases (KDD'97).*

B. R. Kent. 2013. Visualizing Astronomical Data with Blender. *Publications of the Astronomical Society of the Pacific* 125 (6 2013), 731–748. Issue 928. https://doi.org/10.1086/671412

X. Li, A. Kuroda, and H. Matsuzaki. 2016. Polyspector™. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 109–111. https://doi.org/10.1145/2984751.2985720

J. Lowe and M. Matthee. 2020. Requirements of Data Visualisation Tools to Analyse Big Data: A Structured Literature Review. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12066 LNCS (2020), 469–480. https://doi.org/10.1007/978-3-030-44999-5_39/FIGURES/3

I. S. MacKenzie and C. Ware. 1993. Lag as a determinant of human performance in interactive systems. *Conference on Human Factors in Computing Systems - Proceedings* (1993), 488–493. https://doi.org/10.1145/169059.169431

J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. Hung Byers. 2011. Big data: The next frontier for innovation, competition, and productivity. (2011).

L. Marchetti, T. H. Jarrett, A. Comrie, A. K. Sivitilli, F. Vitello, U. Becciani, and A. R. Taylor. 2020. iDaVIE-v: Immersive data visualisation interactive explorer for volumetric rendering. *arXiv* (2020), 1–4.

M. M. Masiane, A. Driscoll, W. Feng, J. Wenskovitch, and C. North. 2019. Towards insight-driven sampling for big data visualisation. (2019). https://doi.org/10.1080/0144929X.2019.1616223

T. McReynolds and D. Blythe. 2005. *Scientific Visualization*. Elsevier. 531–570 pages. https://doi.org/10.1016/B978-155860659-3.50022-6

MDN Web Docs. 2024. WebSockets API. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API Accessed: 2024-05-20.

A. Merv. 2011. Big data. Teradata Magazine Online, Q1.

T. Muhlbacher, H. Piringer, S. Gratzl, M. Sedlmair, and M. Streit. 2014. Opening the black box: Strategies for increased user involvement in existing algorithm implementations. *IEEE Transactions on Visualization and Computer Graphics* 20 (12 2014), 1643–1652. Issue 12. https://doi.org/10.1109/TVCG.2014.2346578

J. P. Naiman. 2016. AstroBlend: An astrophysical visualization package for Blender. *Astronomy and Computing* 15 (4 2016), 50–60. https://doi.org/10.1016/j.ascom.2016.02.002

E. Nash, G. Edwards, J. Thompson, and W. Barfield. 2000. A Review of Presence and Performance in Virtual Environments. *Int. J. Hum. Comput. Interaction* 12 (05 2000), 1–41. https://doi.org/10.1207/S15327590IJHC1201_1

N. Nilsson, R. Nordahl, and S. Serafin. 2016. Immersion Revisited: A Review of Existing Definitions of Immersion and Their Relation to Different Theories of Presence. *Human Technology* 12 (11 2016), 108–134. https://doi.org/10.17011/ht/urn.201611174652

R. Norris. 1994. The Challenge of Astronomical Visualisation. 61 (01 1994), 51.

D. Punzo, J. M. van der Hulst, J. B. T. M. Roerdink, J. C. Fillion-Robin, and L. Yu. 2017. SlicerAstro: A 3-D interactive visual analytics tool for HI data. *Astronomy and Computing* 19 (April 2017), 45–59. https://doi.org/10.1016/j.ascom.2017.03.004

R. Rosenbaum and H. Schumann. 2009. Progressive refinement: more than a means to overcome limited bandwidth, Katy Börner and Jinah Park (Eds.). *Visualization and Data Analysis 2009* 7243, 72430I. https://doi.org/10.1117/12.810501

L. Rosenblum, R. A. Earnshaw, J. Encarnação, A. Kaufman, S. Klimenko, G. Nielson, F. Post, and D. Thalmann. 1994. Scientific Visualization—Advances and Challenges. (01 1994).

M. Sanchez-Vives and M. Slater. 2005. From presence to consciousness through virtual reality. *Nature reviews. Neuroscience* 6 (05 2005), 332–9. https://doi.org/10.1038/nrn1651

B. Shneiderman. [n. d.]. The eyes have it: a task by data type taxonomy for information visualizations. *Proceedings 1996 IEEE Symposium on Visual Languages*, 336–343. https://doi.org/10.1109/VL.1996.545307

B. Shneiderman. 2008. Extreme visualization. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 3–12. https://doi.org/10.1145/1376616.1376618

J. Steuer. 2000. Defining Virtual Reality: Dimensions Determining Telepresence. *Journal of Communication* 42 (07 2000). https://doi.org/10.1111/j.1460-2466.1992.tb00812.x

I. E. Sutherland. [n. d.]. The Ultimate Display. http://www.cee.hw.ac.uk/courses/5ig2/1/ultimate_display.html

R. Taylor. 2015. Frelled: A realtime volumetric data viewer for astronomers. *Astronomy and Computing* 13 (11 2015), 67–79. https://doi.org/10.1016/j.ascom.2015.10.002

R. Taylor. 2016. Visualising three-dimensional volumetric data with an arbitrary coordinate system. (11 2016). https://doi.org/10.1088/1538-3873/129/972/028002

CARTAvis Team. 2023. fits2idia: FITS to IDIA Image Conversion Tool. https://github.com/CARTAvis/fits2idia. Accessed: 2024-09-19.

E. R. Tufte. 1983. The visual display of quantitative information. (1983), 197.

J. W. Tukey. 1977. *Exploratory Data Analysis*. Addison-Wesley.

M. van Zyl. 2022. *VRDAVis GitHub repository*. https://github.com/Michaelazzz/VRDAVis

K. Walsh and S. Pawlowski. 2002. Virtual Reality: A Technology in Need of IS Research. *Communications of the AIS* 8 (03 2002). https://doi.org/10.17705/1CAIS.00820

WebRTC. 2024. WebRTC: Real-Time Communication for the Web. https://webrtc.org/ Accessed: 2024-05-20.

D. C. Wells and E. W. Greisen. 1979. FITS-a flexible image transport system. In *Image processing in astronomy*. 445.

C. M. Wittenbrink, M. Meißner, H. Pfister, and R. Westermann. 2000. Volume visualization and volume rendering techniques. https://www.researchgate.net/publication/228557366

I. Wohlgenannt, A. Simons, and S. Stieglitz. 2020. Virtual Reality. *Business Information Systems Engineering* 62 (10 2020), 455–461. Issue 5. https://doi.org/10.1007/s12599-020-00658-9

C. Yang, Q. Huang, Z. Li, K. Liu, and F. Hu. 2017. Big Data and cloud computing: innovation opportunities and challenges. *International Journal of Digital Earth* 10 (1 2017), 13–53. Issue 1. https://doi.org/10.1080/17538947.2016.1239771

J. S. Yi, Y. A. Kang, J. T. Stasko, and J. A. Jacko. 2007. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics* 13 (11 2007), 1224–1231. Issue 6. https://doi.org/10.1109/TVCG.2007.70515

H. Zhao, H. Zhang, Y. Liu, Y. Zhang, and X. Zhang. 2017. Pattern discovery: A progressive visual analytic design to support categorical data analysis. *Journal of Visual Languages Computing* 43 (12 2017), 42–49. https://doi.org/10.1016/j.jvlc.2017.05.004

A Participant 1

Interviewer: Do you have any thoughts? On the experience or just any feedback?

Participant 1: I mean, yeah, I guess. am I allowed to put on my i-Davie hat.

Interviewer: It's actually quite beneficial that you compare it to i-Davie?

Participant 1: Well, I think one of the things that people notice right away with i-Davie. I can't speak for other people. But when it comes to the, I guess what with this kind of software, it comes down to the visualisation and plus the interactivity. So this feels like it's really nice that I can actually I get that self contained standalone visualisation. That's awesome. It just the interactivity, I feel it's not there. Where it's like, yeah, I can draw the box, the box felt a little bit funny. Like, I didn't feel like it was like one-to-one, like going through the cube. Like, I don't know if it was growing from the middle or? Yeah, whichever people have requested for i-Davie, but I don't like that. I think it's, it can be convenient for certain things. But others, I think it's actually off-putting because it's almost like I don't know, I mean, because I've always used like, Photoshop and things like that. Be aware, I'll never have a situation where I click and hold and then uh, then the selection box gets bigger and smaller, right? I just I select this, and I still and it's like one to one what I want to have selected, right. And it's maybe a little bit inconvenient for us. I want to select around maybe an object of interest, which astronomers do. And the thing to kind of notice. And then it kind of goes. And then that being that, maybe that could be a separate mode, I don't know. That's not the only interactive kind of element that makes it feel bad, that the cube seems kind of distant and I'm not actually interacting with it. And I think that's especially the case where I don't actually have a cursor on my hand to show like, Okay, this is the value of that point. There's value in that point. So I think that would definitely be kind of a earliest add on, I guess to this is adding the real time kind of interaction of like, this is the value of that point. And that is the what he called the coordinate to that point. And then I can move it through the cube and then actually find where I'm at because otherwise it's a little bit yeah, even though it is the 3D space. I don't feel like the the the the immersion is kind of gone. The presence, I guess is also kind of gone.

Interviewer: That drag from the middle function of the selection box was just the easiest to implement at the time.

Participant 1: That makes sense to use. Because it probably for in the case of astronomy is probably a lot more useful. Because usually astronomers are worried about individual spots and they want to select that region and then

just get a designated kind of radius around that to crop to it. That makes sense. But, yeah, and then the menus did feel a little bit funny. Yeah, because I don't know what. Yeah, when you're dealing with all these, like very low level kind of libraries, then you need to add into your seeing, versus like us where we deal with Unity, where a lot of things are built for us. And it's definitely a challenge. But I think with menus, it comes down to a lot of the time, too, if you can emulate what people are used to. So when I want to move a menu, in, in Windows, I need to grip the side of the mouse and move it around, I just literally click the ribbon at the top and drag it around, right. And in VR, that makes sense for me as well. You just have to maybe make it a little bit fatter, that banner at the top just to make sure they can drag it around. Be I think other than that, it's it's stunning. To me, the main thing is that is the volumetric rendering that you got in on the standalone device. It's incredible. And then the performance didn't seem that bad, it look like didn't stutter at all.

Interviewer: The stutter was awful. Like when I took it to Taiwan, it was so bad because the UI library I was using was running additional loops, like every frame. So it was just making the computation of the scene just astronomical. As soon as I took that library out it ran so smoothly. So then that's why they have janky menu, because that's what is the trade off for performance.

Participant 1: And that's not a menu said like a UI system that's made for VR.

Interviewer: No it's not specifically made for VR, it's just canvases on plains, but I did manage to get chartJS onto one of those. So that's a dynamic chart that changes with the data. Which I'm also very happy with.

Participant 1: Yeah, I mean, at the end of the day, it's it's quite the task you add without using Unity. Because for us, it was just dragging things in and they were kind of tested to work already versus doing a lot of things from scratch on your end. They I think it's certainly a good, a good start.

Interviewer: What do you think of the workflow integration? Successful? How do you feel about it?

Participant 1: So between the desktop?

Interviewer: Yeah, so if you had to start with the desktop, and then move on to the VR side of the headset? Do you find that if the astronomers want to look at a cube in VR, they typically just start from the VR?

Participant 1: You know, I mean, I definitely would think that like a similar system, where I'm selecting the file from the menu on the desktop, and then going

into VR to do what VR is good for. And that's this to see things in 3D. That's to see things in 3D, navigate around the cube. I don't want to have to select a cube from within VR unless necessary, because sometimes, maybe I just made one little mistake. Maybe I'm setting parameters on the cube or something and I want to go back really quick. There wouldn't be or maybe there was a second cube. And it'd be nice to load up really quick. But that's not worth putting a whole file selection. And I think basically keep VR to a VR is good at and that especially that's in both directions. So basically sending things, though, so before I put on the VR, and then after I take off the headset, what am I getting out of it? I mean, there I didn't. I didn't feel like I was producing anything yet. And it's difficult because I'm seeing stats, I'm seeing the graph. And maybe downside point to an area. But I mean a lot of these numbers. So I don't know if I'm exporting or, or putting, like what actually productive I'm getting out of the session, I feel because there's not a other direction thing yet. And that's kind of lacking. So in terms of being in the workflow, it's more of a work endpoint. But yeah, it's good that I can select the file from the desktop and then go in. And then I just have it as well. It's really nice that when things did crash, it was really straightforward just to do just to refresh and then that was actually quite nice. That things didn't just like it catastrophic. And then you have to ask you who's the expert at it. To grab the cube then fix it for me like it actually was quite straightforward. Just reload the scene.

Interviewer: Okay, that's really nice. I was like, yeah, it's just a lot of like, polish stuff. Additions like next steps. Yeah, that's, that's what I'm getting from the feedback. So I'm very happy because I'm like, Oh, this is this is garbage Oh, that's great feedback. Okay. Thank you so much. That's great feedback.

B Participant 2

Participant 2: That was quick. Yeah, I think yeah. If you because of the of the selection, that is a shape that is not attached to the cube. It's also because I'm so familiar with how i-Davie works, right? You have all these other things. So it's actually what I expect to do I guess. Yeah. So so maybe that's why I found it a little bit counter-intuitive, but it's also because I'm used to the mapping already. But it's working in the transfer minimum data search its a matter of perfecting it a little but there is something there.

Interviewer: It does require more polish, I do completely understand what you mean by the controls. They are a little weird, and more thought needs to be put into it which will be a done though the iterative development process.

Participant 2: But I didn't have any problem with motion sickness, maybe I want it to be forgiven with respect to...there was not a lot of jitter. So I guess it just didn't do it that it didn't like go crazy.

Interviewer: At a stage it was jittering to the point where I was getting sick. So that was that was because of the UI library we were using was adding extra rendering frames within the rendering loop. each frame has a rendering loop. So every rendering function it was adding more loops inside the render function. So it's just crazy, but we counteract the jittering by scaling down the steps it takes with the raycast shader. So the loop would just stop. We just tried to keep it as smooth as possible, because it is a comparatively lower powered hardware compared to the computer you use to run i-Davie on.

Participant 2: So but then the you're, you're using the headset computer to do some of this calculation, or all of them.

Interviewer: Yeah, all of them. An I just wanted to ask you some questions. It's just to make sure I don't forget anything. So obviously the struggles you encountered was with actual interacting with the cube.

Participant 2: Yeah, yeah. But as I said, because I'm used to interacting with the cube in a different way. Also, my, my way of reaching to the wrong bottom and need to remember the now okay, this is the trigger. And this is the other one says different words.

Interviewer: Yeah, its just a learning curve.

Participant 2: Yeah. But it's not difficult to you know, if I wasn't used to the other one, I would probably just do it. It's just that I'm used to do something else.

Interviewer: You know, obviously, you've seen data like this being displayed in this way in VR. So yeah, what do you think about?

Participant 2: Well, I think our right marching is better to see the definition of the data. This one, it's really, almost, I can't wait. You know, when you see those sci-fi movies of the 70s, where there are the space travel, things of those lines appearing and disappearing in that world, believe me, as I perceive that vision, so you can't really see much of the data details. So that's, that is a little bit limiting in that respect. But again, I'm so used to see how we do it a different way that I guess I'm a little bit less forgiving than others, maybe you know.

Interviewer: But that's absolutely fine. What did you think of this, this workflow, moving from the laptop, to the headset?

Participant 2: I'm thinking it would work? I mean, I think I'm a little bit biased because I work in this project (i-Davie). But, yeah, I think because of the size of the data we will have to deal with, I think its going to be important to have capability of reducing the data into the data of interest before you immerse yourself. This is of course with the idea that people will try visualise a cube of an entire survey in one go and go to the single source to analyse. Depends really on what type of science...of pipeline people will use. Which is a new thing. We don't know really because we've never had the capacity to do that. So I don't know. Having both sides of the thing, because VR does not good to do everything perfectly because it's so different to the screen.

Interviewer: I don't thing the system displayed its capabilities of being able to drill down into the cube. It's a little bit funny. A little but temperamental. Sometimes you get a good piece of the data. Which is something I would like to correct. An just one more, how does it compare to i-Davie?

Participant 2: Well, how tricky, I mean, I really such a mature thing right now. I mean, you can see this is still experimental. I like it better how we grabbing things in the i-Davie, to be honest, I find the more intuitive or we have put ourselves into understanding how people and also maybe now I decide I'm so used to that, that I find it easier. So the grabbing of this, this action of grabbing space, instead of the trigger, I find it more intuitive because it's really grounding in the concept. So for example, the mapping of the control if you want the rendering also, to be clear, in i-Davie because of that, and of course, I mean, i-Davie has all the analytics. But I think the i-Davie problem is dealing with big data, this algorithm that works, you can have the marriage of the two. That's the idea I guess

C Participant 3

Interviewer: So I'm just going to ask you a few questions about how you feel about the experience and stuff like that. Okay so what do you think was the main point that you struggled with, well obviously the headset. Did you feel like you struggled with anything?

Participant 3: No, I think you get used to the headset in time and unfortunately that's not something you can control. But it was fine it was very quick, who you use headsets will get better the more you use it. But no specific thing. Obviously it would be useful to have more features, especially the contrast, because even for just looking at it, it was hard to see the emission lines.

Interviewer: So since you have used i-Davie before, how do you think the two

systems compare?

Participant 3: Its difficult to comment. I have used it (i-Davie) with much larger cubes, so it would take time to load. But I don't know how that compares, and it also has more features.

Interviewer: i-Davie is definetely the more mature software.

Participant 3: But it's nice that it can be opened from anywhere, that's very very good. Because then you don't have to come to a machine all the time, you are just at your computer and you can use it. Yeah that's really nice.

Interviewer: That's great feedback. Obviously you've looked at data-cubes in VR, so how did you find the visualisation, the quality of the visualisation?

Participant 3: I think I can only say that more if there's, I mean, I just felt that contrast could have been better. For example visualising the high density gas, low density features, and the background. I mean its good already.

Interviewer: Yeah, this where we find the things we can work on. And than you so much for your feedback.

Participant 3: And it is also nice to change the scale like the maximum and the minimum, but that us based on what you want to look at right? Like really faint features or depending on what I want to look at, an option to change the values.

Interviewer: Yeah, so basically more features that's really great feedback to hear.

D Participant 4

Interviewer: Now we are just going to move onto some free-form questions about what you experience. Okay so obviously you've worked with VR astronomy data visualisation software before. Do you have any feedback on the experience as a whole?

Participant 4: I mean very impressive, I think i was saying the hardest part for me was turning the data. Aside from that it seems its quite clear to see what's in the cube and I did find also you know when the cube goes behind the menu then one has to find a way to move around that.

Interviewer: Yeah so some user interface issues which are common.

Participant 4: And then I guess the fact that the crop cube sticks around.

Interviewer: User interface issues, completely understandable. And then what do you think of the workflow, of like having a cube open in CARTA you just go to the VR headset, do whatever you need to do in VR and then just go back to the computer when you're done and all the data from VR is back on the computer. How do you feel about that workflow?

Participant 4: That sounds like that would be great, because I mean CARTA to have the slices in CARTA and just looking at it like that it's sometimes hard to get a good visualisation of what the cube is doing.

Interviewer: It seems like you've had a very positive experience with VRDAVis. Most of the time these post-test interviews determine what the major issues are. I see you didn't find any major issues with the system itself. It's value proposition is still there even with a clunky interface.

Participant 4: I think it would be a very helpful thing to have. As a way to have a quick look in VR for just regular astronomy use.

Interviewer: And then one last question, since you have used i-Davie before how do you think the two systems compare?

Participant 4: I mean, so as I say just general interface, turning and manoeuvring the cube is more streamlined in i-Davie and I mean I think also the general clarity of the data in the cube. I don't know much about the technicalities of how that works but it did seem visually clearer on i-Davie, I don't know why that would be.

Interviewer: The reason for that is i-Davie run on a higher powered machine compared to this headset, so it can load in more data, so the resolution of the cube you are looking at looks a lot nicer. This tries to counteract having to load in so much data for performance by loading in pieces. This is why the crop functionality is a core feature. So like as you drill down the data becomes higher resolution.

Participant 4: I mean, I think for the purpose of having a quick look at something instead of having to load it up in a whole i-Davie, it would serve its purpose.

E Participant 5

Interviewer: So, a few questions. While you were using the system did you struggle with anything?

Participant 5: The most annoying thing is probably initially getting a handle on zooming in and zooming out, and moving the menu away. But after a while

you kind of get used to it.

Interviewer: What did you think of the quality of the visualisation? The big floating cube.

Participant 5: It was fine. Was it a masked image?

Interviewer: Yeah, it might just be. Other ones I looked at were quite noisy, that one is masked.

Participant 5: It was great at looking at the masked one, I would like to see what a unmasked one would look like.

Interviewer: How do you think it (VRDAVis) compares to the i-Davie system?

Participant 5: I will say the actual interface, like getting into the VR thing was a lot simpler than i-Davie. The one time I used i-Davie it took 10 minutes to open up the cube and get into the platform because there was an issue with the FITS file. Other than that it was quite chilled. I obviously didn't do everything I did with i-Davie on this bad-boy.

Interviewer: Okay, great. An then what did you think about the integrated workflow between the laptop/desktop and the headset.

Participant 5: That was cool. I liked that a lot actually. I like the idea that i could randomly put on the headset and look at something. Yeah, that was great.

F Participant 6

Interviewer: So I'm just going to ask you a few questions about your experience that you had. Do you feel like you struggled with anything in particular.

Participant 6: Not really. Just the crop mode at the beginning (they are referring to accidental activation of the crop mode at the beginning of the evaluation).

Interviewer: So just some interface errors and stuff like that.

Participant 6: I think I clicked without realising. It was through the cube, but that was fine.

Interviewer: What do you think of the proposed workflow. Starting from the laptop and transferring the state to the headset?

Participant 6: That's quite nice. It seemed to be quicker than loading it into the headset, using i-Davie for that I know takes a while. So it is quite nice to be

able to jump between them.

Interviewer: An then, since you have used i-Davie before, how do you think the two systems compare?

Participant 6: I found them pretty similar, and I presuming this is like a basic version because I know i-Davie has moment maps and crop the cubes but quite similar.

Interviewer: How did you find the quality of the visualisation?

Participant 6: Good, good. Like I said, just like i-Davie, pretty good. And then also with the low quality cube, I could discern the different intensities, the emissions in the cube, and depth,

G Participant 7

Interviewer: Did you feel like you were struggling with anything in particular while you were using the system?

Participant 7: I think the only thing was the crop button was in my line of sight. The cube was between me and the crop button. But it would make sense for me to move the cube to the side, move here and then crop it again. That was the only thing.

Interviewer: So how do you feel about this proposed workflow that this system implements?

Participant 7: I think it's great. I guess once your cube is uploaded to the system its pretty seamless.

Interviewer: Great! And then you have used i-Davie before?

Participant 7: Yes.

Interviewer: So how to you think the two systems compare to each other?

Participant 7: Um, well, so when I used i-Davie I didn't upload anything myself. So usually I would give my cube to be opened up. But I mean it would cut out the middle man. If there was a way for me to take the headset off, use it an put the headset on and then manipulate and maybe load a different cube. Yeah I can see the benefits in this.

Interviewer: The only extra bit of admin around loading a cube onto this system, they just have to pre-processed before you can use them.

Interviewer: How did you find the quality of the visualisation in the VR space?

Participant 7: Yeah it was great, crystal clear.

Interviewer: No jittering and stuff like that.

Participant 7: No. I mean when I was scaling it, it wasn't a smooth motion. But the data itself was fine.

Interviewer: Okay great, fantastic. That's all the question I have for you.