Remote Visualisation of 3D Astronomical Data

Honours Literature Review

MIVHAELA VAN ZYL VZYMIC015, Department of Computer Science, University of Cape Town, South Africa

Astronomical data presents certain challenges that affect the approaches we take when developing a system that can visualise the data [8]. Firstly, there is not a single dominant format which means attempting to make a generic system would be difficult, a single format must be selected to base the system on. There is also the aspect that astronomical data has a low signal to noise ratio as well as a high dynamic range. It becomes difficult to separate the data from the noise and to normalise the data to within a range that can be visualised. Astronomical data uses dimensions in a different way to typical two and three dimensional spatial datasets, and maps different data types to axes. Correctly mapping these axes is imperative to accurately visualising the data. Possibly the greatest challenge is the size of the data that is produced by astronomy recording instruments. High resolution data often contains millions of data points and could easily take up the entire storage space of several hundred home computers. This review explores methods for designing systems that can process the data, render the visualisation, and have the visualisation be interactive.

Systems attempting to addressed this problem have been proposed and developed for example SlicerAstro [16], iDaVIE-v [11], Frelled [17], Fips [9] and others that touch on real time visualisation [15] [6] [8] [13].

The remote visualisation of three dimensional astronomy data has many parts that require careful consideration taking into account the nature and challenges presented by astronomical data especially the size of it. Starting from preprocessing the data so that it is sufficiently minified so that it can be rendered effectively and as fast as possible while maintaining as much detail as possible. Then transferring the rendering over a network like the internet, has to take into account the latency of transferring the data and how the data will be received by the client. While presenting this data to the user in a manner where they can intuitively interact with and manipulate it to extract knowledge from the dataset.

1 INTRODUCTION

1.1 Background

The astronomy field conducts their research by observing the visible universe and collects data through telescopes and satellites. These data collection methods generate extremely large amounts amounts of data. Scientists wanting to study the data need to interact with and explore the collected data which is done through the visualisation of the different parameters contained within the data. Through the interaction and manipulation of the data scientists are able to make discoveries and gain insights into our universe.

1.2 Motivation

There is a need within the astronomy community for a system that can reliably produce and accurately portray data while accommodating the complexities associated with processing and rendering astronomical data.

The main portion of the data's processing and visualisation would be done by a remote system which would give a wider range of people access to the data regardless of the computational power of their local systems. It also reduces the need to transfer the data between local systems and would remove obstacles from the scientists path in being able to interact with the data and study it effectively.

1.3 Overview

This review explores concepts in preprocessing, rendering, visualisation and interaction in the context of their application to remote visualisation of three dimensional astronomical data.

Section 2 discusses the different ways in which three dimensional data can be visualised and how they perform when representing astronomical data.

Section 3 explores the different preprocessing techniques for minifying and grouping data to reduce the computational overhead during rendering.

Section 4 discusses the different methods in which to render a three dimensional model and how these models can be rendered in real time as well as remotely.

Section 5 discusses methods for interaction with the produced visualisation.

2 DATA VISUALISATION

Data needs to be visualised in order for it to be interacted with in an intuitive way. There are a multitude of ways in which to represent and render volume data. Similar techniques for data visualisation for astronomy are used in the visualisation of medical volume data because both fields have to contend with how to represent large amounts of data [5] [10]. The following subsections will elaborate on different visualisation techniques and their ability to render volume data.

2.1 Points

Point representation of a dataset simply plots each data point as a fixed width pixel to the picture plane. It is the most straightforward way of representing data but it is limited by the available resolution of the data. This technique is not very effective for astronomical data because simply plotting the data as a three dimensional scatter plot makes it difficult to derive meaning from the data visualisation. For example in figure 2 the top left quadrant when compared to the bottom right quadrant is much less semantically comprehensible.

2.2 Splats

Splat visualisation makes use of small textures that always point towards the camera regardless of the orientation of the scene [6]. Combining splats produces an visualisation that is much like volume rendering.

2.3 Isosurface

Polygon based or surface rendering is a technique used to represent volume data and it functions by extracting polygons from three dimensional data. Part of the process of modelling the data is discerning where the boundaries of the objects are and defining a skin or boundary for the region in the data it occupies [5].

This technique does not represent astronomical data very effectively as volume rendering. There are many blurred and overlapping

boundaries and therefore this technique does not portray a full representation of the data because of the need of binary boundaries. It is less effective at giving a global view of the data than volume rendering [6].

2.4 Volume Rendering

Volume rendering represents three dimensional data as voxels which can be translated from data points in the dataset and projects them into the picture plane [5] and does not directly deal with surfaces [12]. The colour and opacity is computed for each voxel and classifies them in terms of what proportion they are of each object class. It does not require the explicit classification of binary surfaces to define the data as objects and non objects like in isosurfaces. Meaning is added to the data through colouring the rendered points based on a parameter of the data like temperature [1]. This rendering method produces a model which has a semi transparent gel appearance and is suited for displaying weak or fuzzy surfaces which is a much better way of representing astronomical data. This gives volume rendering the ability to render both the external surfaces and the interior three dimensional structures 1 which would have otherwise gone unnoticed [6] [14] [12].

However, the rendering time grows linearly with the size of the dataset. We encounter problems when the dataset reaches a size where the computational overhead for rendering all the voxels in the scene becomes too great as all the voxels in the scene participate in the rendering. It ia also difficult to add textures to the model's surface because a texture requires a defined surface for it to be wrapped effectively. Volume rendering does not have the necessary defined surface for the textures. The solution to this is to project the textures through the space and onto the model [5].

3 DATA PREPROCESSING

When working with very large data sets that can range up to terabytes in size and then given the task of rendering a three dimensional model using the dataset, some visibility culling has to take place where large parts of the scene are rejected before the visible surfaces are determined to reduce the rendering complexity [4]. Voxel scenes are composed of a cubical grid of three dimensional pixels (voxels), each voxel is either empty or contains some information [18].

The following subsections will discuss various preprocessing methods which incorporate visibility culling and voxel representation.

3.1 Mipmaps

Mipmaps for two dimensional images are pre-calculated levels of images with each subsequent image having a lower resolution than the previous one. The resolution is decreased by a factor of two from one level to the next. Higher resolution images are used to represent objects that are closer to the camera and the objects decrease in resolution as their distance from the camera is increased.

It is effective for minifying the level of detail while maintaining image quality of a three dimensional rendering. Astronomical data can exceed millions of data points so preprocessing the dataset so as to not render detail that cannot be seen on the screen is a substantial reason why this technique is considered. By preventing unnecessary processing and rendering, it reduces computational stress on the GPU and CPU, speeds up rendering times and requires less storage space [3].

Mipmaps can also be used for three dimensional models where instead of resampling a two dimensional layer the entire texture volume is filtered down to an eighth of the original using octrees which are discussed next section. It does this by averaging the eight adjacent texture voxels on one level down to a single voxel on the next. Three dimensional mipmapping has the same benefits as two dimensional mipmapping. This approach is favourable for volume rendering 2 of large volumes of voxels [10].

3.2 Sparse Voxel Octrees

Octrees recursively divide data up into eight smaller cubical subvolumes, also know as octants, up to the desired level. Sparse voxel octrees exploit sparsity in data by omitting sections of the data from the octree that are essentially empty spaces. These empty spaces are not stored in memory [18].

4 RENDERING

To effectively render astronomical data attention needs to be paid to how each rendering technique produces the model because this has an effect on how the data cube can be visualised. The sheer size of the models [7] [15] makes it difficult to render them in real time and models must also be able to be transferred to the user's local system which adds latency. Therefore efficient rendering techniques are required to render a usable model as fast as possible. The subsections will discuss the different techniques for rendering three dimensional models, how these models can be rendered in real time, and how they can be rendered remotely.

4.1 Rasterisation

Rasterisation is currently the most widely used rendering algorithm and is also know as object order rendering. Objects that are going to be rendered are projected sequentially into an image plane and then they are rasterised into pixels and shaded [4]. It has the best effect in cases where a few triangles cover large areas of the screen and where there are defined objects. This method would not work well with rendering astronomical data which is comprised out of many points and does not have explicitly defined boundaries.

4.2 Ray Tracing

Also know as image order rendering, this technique models physical light as straight lines by shooting rays through a pixel grid into a three dimensional scene. The algorithm then determines the ray's closest intersection with the model's surface [4]. It is a more simplistic technique both in theory and implementation than rasterisation, and would be better suited for rendering volume data because the visibility of the model must be determined in a point wise manner.

4.3 Real time volume rendering

Volume rendering of voxel scenes is computationally expensive and makes it difficult to render large datasets in real time and have the data be interactive [17] [16] [10].

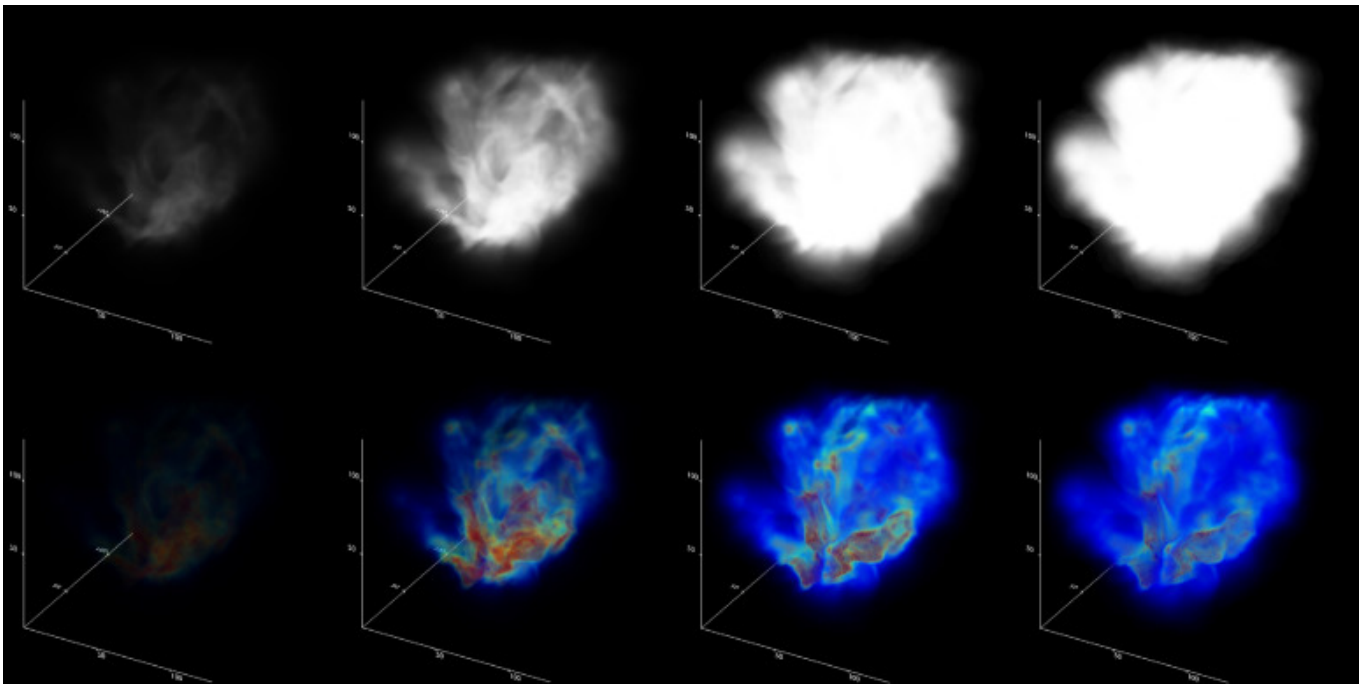

Fig. 1. Astronomical data visualised using volume rendering shows how the classification and colour coding of the individual data points uncover emergent data structures that could otherwise not be seen [17]

Real time volume rendering makes use of a hierarchical pyramid of binary volumes to make up the model. It also makes use of adaptive sampling which was discussed in further detail in section 3.

The volumes are given rank in the hierarchy and their position in the hierarchy determines when they will be rendered. The main volumes that make up the bulk of the model are processed and rendered first and the volumes that contain the detail are rendered after. Figure 3 illustrates how the volumes are loaded and rendered using progressive volume rendering.

The HDF5 data format is stored in a manner the same as a file directory, it is also commonly used to store astronomy data. This format sorts the data into a hierarchy and therefore reduces the need for the dataset to be processed. Storing data in a hierarchical format like HDF5 can potentially cause rendering techniques to be effective as it can facilitate the identification of coherent regions within the dataset.

4.3.1 Remote Rendering. Remote rendering of data cubes as opposed to visualisation on local system has to be put into consideration due to the fact that the data easily exceeds the size of a typical local system's memory [15]. It would be desirable to do the computing of the data on a remote system that has sufficient power and capacity and would provide the wider astronomy community with a low cost visualisation service [2] as well as reducing the need to transfer data between systems. The user would request a visualisation of a dataset from the remote host and the remote host would send the visualisation to the clients computer [6]. Figure 4 outlines the algorithm for remote rendering.

Real time interaction with large datasets over the internet is a challenge because one not only has to account for processing time but also the latency that comes with transmitting data over the internet. A method to reduce the time to render a usable visualisation would be to send the data in a hierarchical format [13]. It is important to note that when rendering the visualisation the order in which the volumes are loaded affects the appearance of the final visualisation.

5 DATA INTERACTION

The visualisation of astronomical data creates a need for interaction, interaction with the data is critical for knowledge discovery. It is necessary to allow the user to manipulate data in a way that does not inadvertently remove details that are of interest to the user while also obtaining an intuitive understanding of the data as well as quantitative results [14] [5].

The biggest challenge for interacting with astronomical data is that the data is so large that the latency for re-rendering the model results in the framerate dropping to a point where interacting with the data becomes virtually impossible [16].

Frelled [17] uses the software Blender that has all the desired characteristics for smooth and effective interaction because it is designed to manipulate and edit detailed three dimensional models and performs better than any current astronomical viewers. The user can manipulate the model by enlarging or shrinking it to the desired size as well as allowing it to be rotated on the x, y, and z axes. It would also be desirable to include the ability to change visualisation parameters and dynamic data filtering especially with radio astronomy data that has many parameters that could be visualised

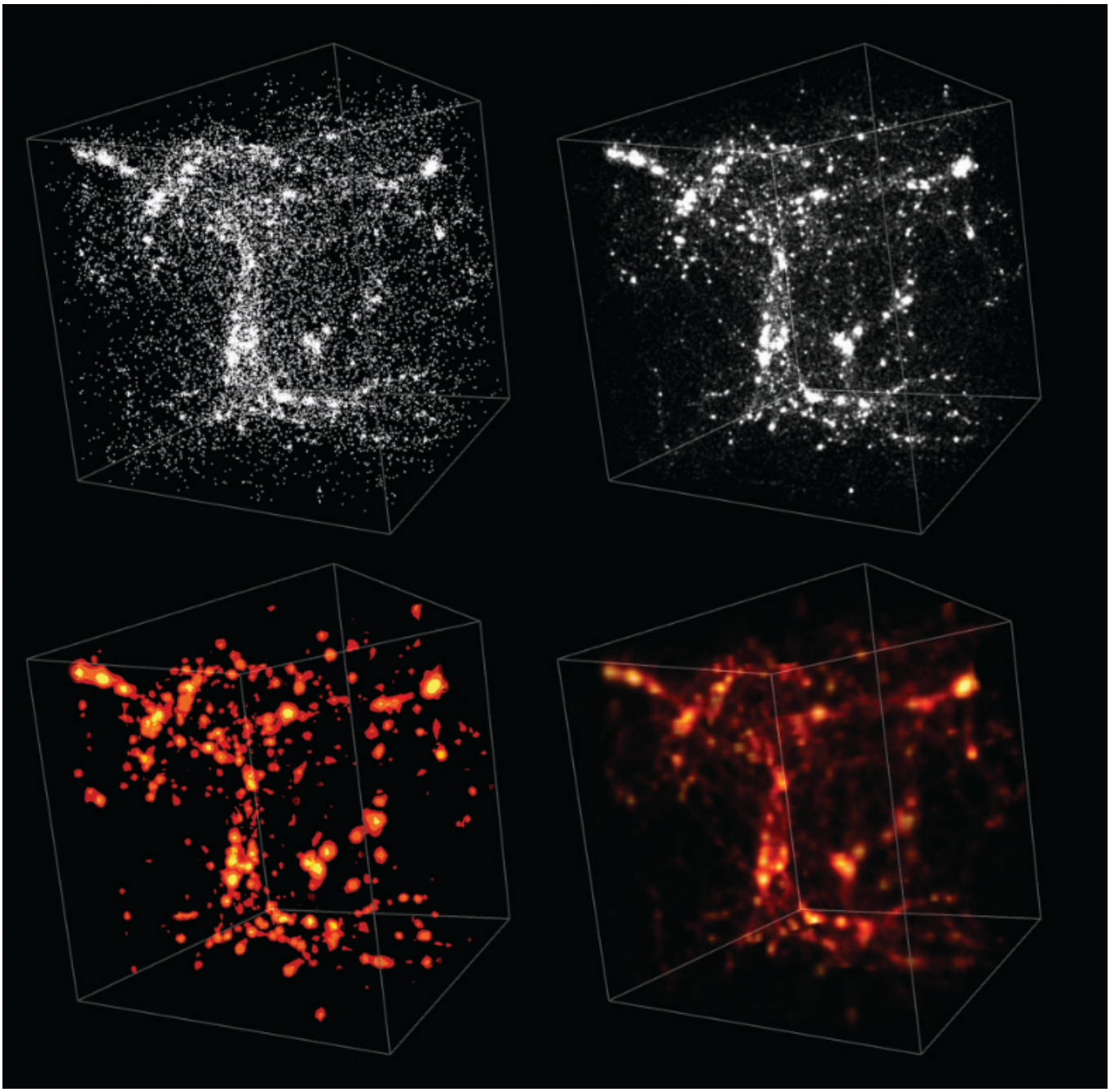

Fig. 2. Different visualisation techniques represent astronomical data [6] Top left: point data, top right: splats, bottom left: isosurface, bottom right: volume rendering.

[6]. This would give the user the ability to intuitively and precisely navigate through three dimensional data.

An example of an astronomical data viewer is iDaVIE-v [11] which is used for data cube exploration. It makes use of the game engine Unity to visualise the data cube and allows the user to interact with the data in an intuitive manner. The use of game engine software enables easy implementation of interaction because game engines are designed to produce programs where interaction is a given feature. Although, all the rendering is done on a local machine and it uses a virtual reality headset to give the user an immersive perspective. When implementing a similar system in a web browser

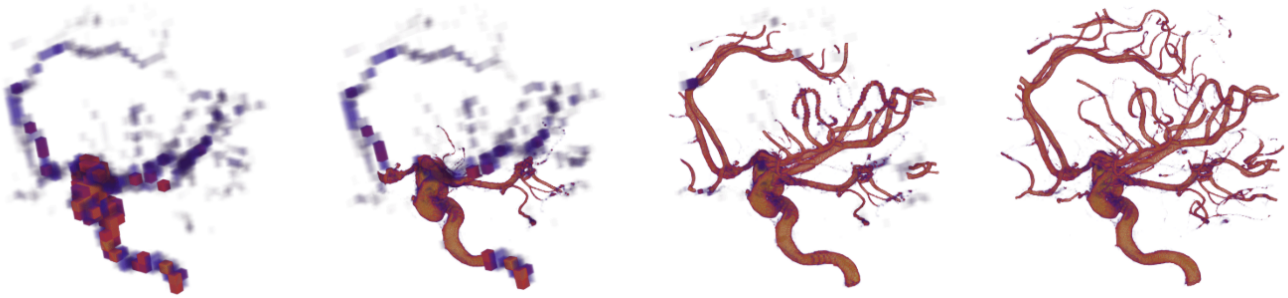

Fig. 3. How progressive volume rendering processes data volumes in a hierarchical manner [13].

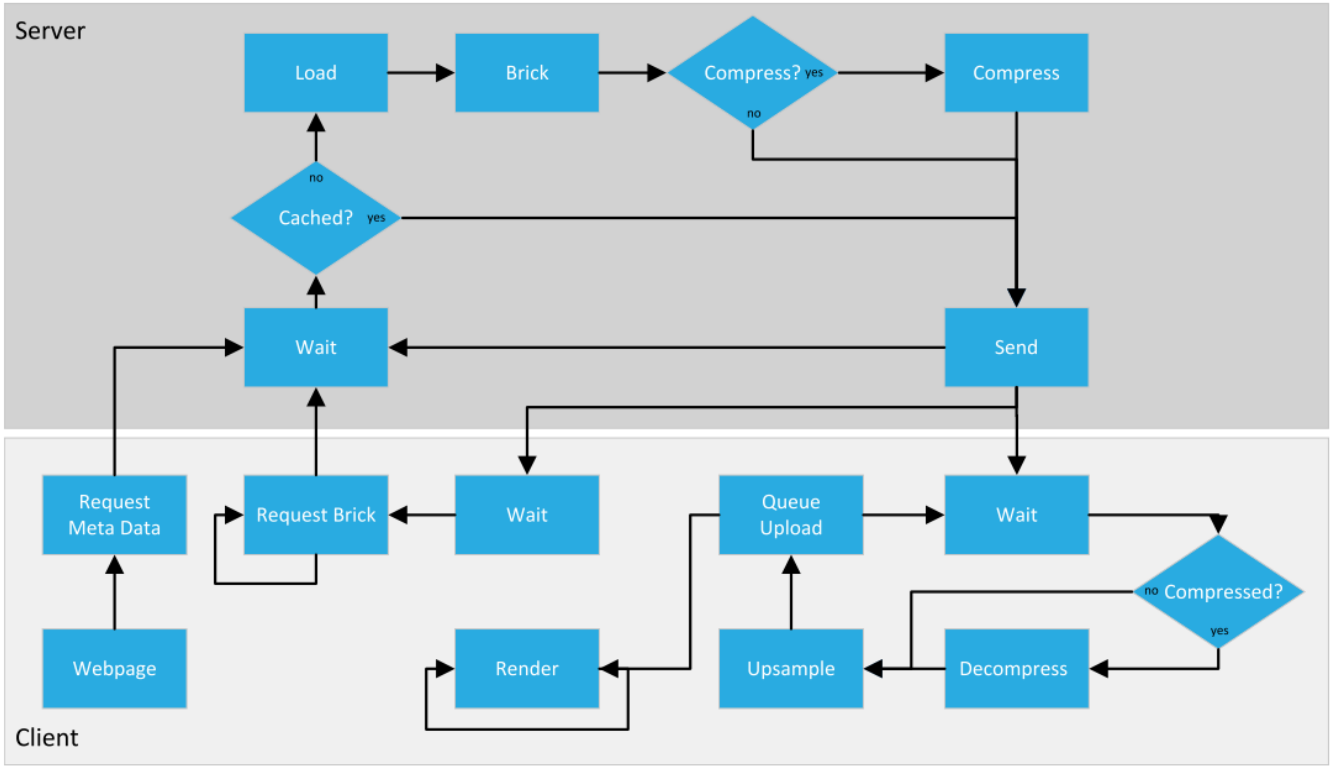

Fig. 4. A possible implementation of an algorithmic pipeline for client-server progressive volume rendering [13]. Stipulating which tasks are handles by the server and which tasks are handled by the client as well as how those different tasks relate to each other.

some degree of interaction is expected and there are tools already available to implement interaction into the client side visualisation.

6 CONCLUSION

In this paper various methods pertaining to how to approach the problem of visualising and interacting with very large astronomical datasets within a web browser were discussed. The size of the datasets indicates that they require a super computer to be able to store, process and render the visualisations. Taking this into account remote rendering becomes the best option to make the data accessible to a wider range of individuals.

Preprocessing of the data before the model is rendered is also required for computational efficiency. The data can be minified using the mipmap method and broken up into volumes and placed into a hierarchical structure which ensures that the most important data is rendered fist and the rest is loaded in a progressive manner as well as ensuring that no unnecessary processing is done. The preprocessing ensures that the rendering and transfer latency is reduced as much

as possible to ensure the client side model is usable for interaction. Sending this data over the internet also requires it to be structured in a hierarchical manner. Progressive volume rendering is a very promising technique for rendering a usable model in a web browser.

Certain visualisation techniques are better suited to the nature of astronomical data, such as volume rendering which is able to reproduce the fuzzy cloudlike nature of astronomical structures and can visualise different parameters of a data points.

How effectively the user can interact with the data is directly related to how quickly the model can be rendered and re-rendering while the user is manipulating it from the client side. If the model cannot be rendered in a fast enough time then the framerate the user sees drops to below a usable threshold. The model becomes unusable because the user must wait for the model to be re-rendered after every adjustment. This would severely hamper the user's ability to extract knowledge from the visualisation which is the primary reason for the implementation of the system.

REFERENCES

[1] Bennett W. Anderson and Robert P. Burton. 1988. Computer graphics curricula: a survey of PhD granting departments. *ACM SIGGRAPH Computer Graphics* 22, 2 (apr 1988), 94–98. https://doi.org/10.1145/47824.47825

[2] David G. Barnes and Christopher J. Fluke. 2008. Incorporating interactive three-dimensional graphics in astronomy research papers. *New Astronomy* 13, 8 (nov 2008), 599–605. https://doi.org/10.1016/j.newast.2008.03.008 arXiv:0709.2734

[3] Bas Dado, Timothy R. Kol, Pablo Bauszat, Jean-Marc Thiery, and Elmar Eisemann. 2016. Geometry and Attribute Compression for Voxel Scenes. *Computer Graphics Forum* 35, 2 (may 2016), 397–407. https://doi.org/10.1111/cgf.12841

[4] Andreas Dietrich, Enrico Gobbetti, and Sung-eui Yoon. 2007. Massive-Model Rendering Techniques: A Tutorial. *IEEE Computer Graphics and Applications* 27, 6 (nov 2007), 20–34. https://doi.org/10.1109/MCG.2007.154

[5] H. Fuchs, M. Levoy, and S.M. Pizer. 1989. Interactive visualization of 3D medical data. *Computer* 22, 8 (aug 1989), 46–51. https://doi.org/10.1109/2.35199

[6] Amr Hassan and Christopher J. Fluke. 2011. Scientific visualization in astronomy: Towards the petascale astronomy era. *Publications of the Astronomical Society of Australia* 28, 2 (2011), 150–170. https://doi.org/10.1071/AS10031 arXiv:1102.5123

[7] A. H. Hassan, C. J. Fluke, and D. G. Barnes. 2010. Interactive Visualization of the Largest Radioastronomy Cubes. *New Astronomy* 16, 2 (jul 2010), 100–109. https://doi.org/10.1016/j.newast.2010.07.009 arXiv:1008.0135

[8] Thomas H. Jarrett, A. Comrie, L. Marchetti, A. Sivitilli, S. Macfarlane, F. Vitello, U. Becciani, A. R. Taylor, J. M. van der Hulst, P. Serra, Neal Katz, and M. E. Cluver. 2020. Exploring and interrogating astrophysical data in virtual reality. *arXiv* (2020). arXiv:2012.10342

[9] Matwey Kornilov and Konstantin Malanchev. 2020. Fips: An OpenGL based FITS viewer. In *Journal of Physics: Conference Series*, Vol. 1525. Institute of Physics Publishing. https://doi.org/10.1088/1742-6596/1525/1/012047

[10] Koojoo Kwon, Eun-Seok Lee, and Byeong-Seok Shin. 2013. GPU-accelerated 3D mipmap for real-time visualization of ultrasound volume data. *Computers in Biology and Medicine* 43, 10 (oct 2013), 1382–1389. https://doi.org/10.1016/j.compbiomed.2013.07.014

[11] Lucia Marchetti, Thomas H. Jarrett, Angus Comrie, Alexander K. Sivitilli, Fabio Vitello, Ugo Becciani, and A. R. Taylor. 2020. iDaVIE-v: Immersive data visualisation interactive explorer for volumetric rendering. *arXiv* (2020), 1–4. arXiv:2012.11553

[12] M Meißner, Hanspeter Pfister, R Westermann, and Craig Wittenbrink. 2000. Volume visualization and volume rendering techniques. (01 2000).

[13] Finian Mwalongo, Michael Krone, Guido Reina, and Thomas Ertl. 2018. Web-based volume rendering using progressive importance-based data transfer. *Vision, Modeling and Visualization, VMV 2018* (2018). https://doi.org/10.2312/vmv.20181264

[14] Ray P. Norris. 1994. The Challenge of Astronomical Visualisation. In *Astronomical Data Analysis Software and Systems III (Astronomical Society of the Pacific Conference Series, Vol. 61)*, D. R. Crabtree, R. J. Hanisch, and J. Barnes (Eds.). 51.

[15] Simon Perkins, Jacques Questiaux, Stephen Finniss, Robin Tyler, Sarah Blyth, and Michelle M. Kuttel. 2014. Scalable desktop visualisation of very large radio astronomy data cubes. *New Astronomy* 30 (jul 2014), 1–7. https://doi.org/10.1016/j.newast.2013.12.007

[16] 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 (2017), 45–59. https://doi.org/10.1016/j.ascom.2017.03.004 arXiv:1703.06651

[17] R. Taylor. 2015. Frelled: A realtime volumetric data viewer for astronomers. *Astronomy and Computing* 13 (2015), 67–79. https://doi.org/10.1016/j.ascom.2015.10.002 arXiv:1510.03589

[18] Remi van der Laan, Leonardo Scandolo, and Elmar Eisemann. 2020. Lossy Geometry Compression for High Resolution Voxel Scenes. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3, 1 (apr 2020), 1–13. https://doi.org/10.1145/3384541