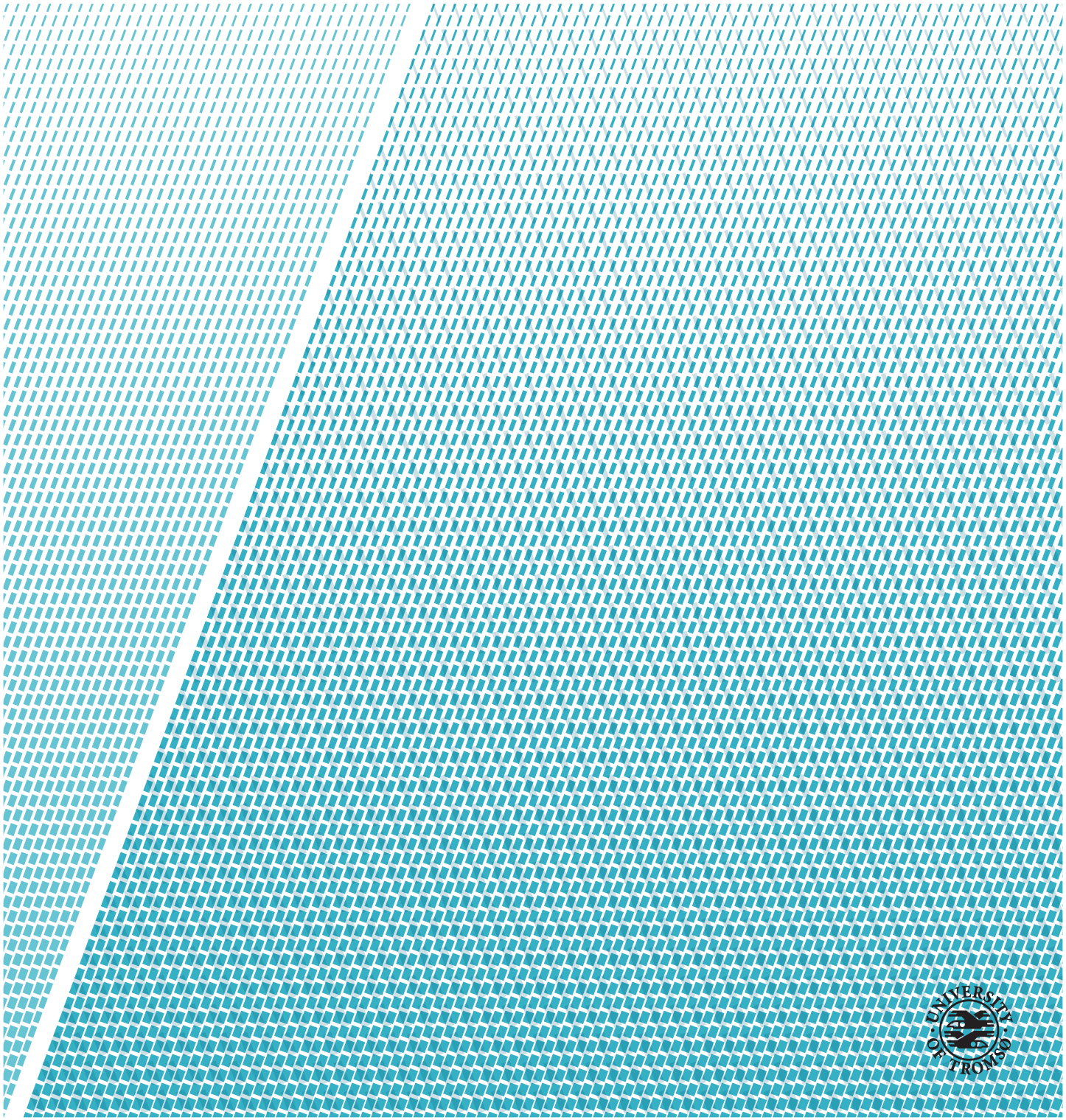
UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Faculty of Science and Technology
Department of Computer Science

GeneNet VR: Large Biological Networks in Virtual Reality Using Inexpensive Hardware

Álvaro Martínez Fernández

INF-3990 Master's thesis in Computer Science November 2020



“Anything created by human beings is already in the great book of nature.”
–Antoni Gaudi

“There are decades of innovations ahead. We’re at the very beginning, where it’s just at the stage where we can bring in consumers [but] there’s so much further to go from there.”
–Brendan Iribe

Abstract

Biological data is often visualized as networks. However, they face problems such as information overload, high interconnectivity and high dimensionality. Existing approaches try to solve these problems by reducing the interactivity in favor of more information or by using specific hardware. This thesis aims to solve these problems using Virtual Reality (VR) and the Oculus Quest, an affordable VR headset. Thanks to the rich interactivity that VR offers and the advancements in hardware, we can reduce the amount of information in favor of more interactivity.

In order to test our hypothesis that Virtual Reality can be advantageous in the visualization of large biological networks, we built GeneNet VR, a prototype of a VR application for the Oculus Quest. As a case study, we used two gene networks from MIxT, a real application with visualization issues. We evaluated the performance and scalability of GeneNet VR and we conducted in-depth semi-structured interviews with several research scientists to evaluate the quality of the application.

The results showed that the performance of GeneNet VR for the datasets from our use case reaches the 72 FPS required by the Oculus' performance guidelines and that GeneNet VR scales well for the largest dataset with 2693 nodes. We also evaluated the performance of GeneNet VR on the Oculus Quest hardware, which also reached the established limit of 72 FPS from Oculus. The Oculus Quest is therefore an affordable option for the visualization of large datasets. From the interviews, we also learned that GeneNet VR has potential as a visualization tool for large biological networks and that is easy to use even for novice VR users. Thus, VR hardware like the Oculus Quest should be considered a competitive solution for visualization tools, as described in this thesis.

Acknowledgements

I would first like to thank my advisor, Associate Professor Edvard Pedersen, for his guidance through this thesis, his constant encouragement, and for the interesting discussions about Virtual Reality and Computer Science. I would also like to thank my co-advisor, Lars Ailo Bongo, for his great knowledge in Bioinformatics, his good criticism, and for all the great insights about scientific writing. Thank you also Vanessa Dumeaux, for introducing me to MIxT and for your great contributions to Biology.

I would also like to thank the collaboration of the several research scientists that participated in the evaluation of my project and that contributed to their knowledge in Biology and Computer Science.

To my close friends Juncal García García, Mireia Nager and Reidar Staupe-Delgado for all the great experiences in Tromsø and for teaching me the value of science.

To Ramsalt Lab, for making it possible for me to study and expand my knowledge in Computer Science.

I would like to thank also my family, for their support and encouragement even from the distance.

Finally, thank you Dominic Ochotorena for your love, for always supporting me, and for being there.

Contents

Abstract **iii**

Acknowledgements **v**

List of Figures **ix**

List of Tables **xi**

1 Introduction **1**
1.1 Challenges and research problem 3
1.2 Proposed solution and contribution 4
1.3 Outline 6

2 GeneNet VR **7**
2.1 Interaction with the network 8
2.1.1 Locomotion 9
2.1.2 Translation of the network 12
2.1.3 Zooming in the network 12
2.1.4 Interaction with the nodes 13
2.1.5 Node relationships 14
2.2 Scalable network in Unity and data structures 14
2.3 Other features of GeneNet VR 17
2.3.1 Filtering information in the network 17
2.3.2 Network morphing 18
2.4 Implementation details 19
2.5 Architecture and design 20

3 MIxT **27**
3.1 What is MIxT used for? 27
3.2 MIxT in VR 29
3.3 Network characteristics 30

4 Evaluation and discussion **31**
4.1 Methodology of experiment setup 31

4.2 For which interactions do we achieve the recommended FPS (72) for large biological networks? 34
4.2.1 Translating the network meets the 72 FPS 35
4.2.2 Scaling the network meets the 72 FPS 35
4.2.3 Selecting nodes the network meets the 72 FPS but with possible bottlenecks 36
4.2.4 Performance discussion 37
4.3 What network properties influence the scalability? 38
4.4 Do we achieve the recommended FPS (72) for large biological networks when using the standalone Oculus Quest? 41
4.5 How do users perceive the visualization of large biological networks in GeneNet VR? 42
4.5.1 How do you perceive the application? 43
4.5.2 How do you perceive the application for pattern finding? 45
4.5.3 What is missing in the application? 45
4.5.4 Discussion 47

5 Related work **49**
5.1 Virtual Reality Chemical Space 49
5.2 BioVR 50
5.3 CellexalVR 51
5.4 BigTop 52
5.5 Unity vs Unreal Engine 53
5.6 VR toolkits and frameworks 54
5.7 VR headsets 54

6 Conclusion **55**

7 Future work **57**
7.1 New requirements based on the interviews 58

List of Figures

1.1 Network view of the MIxT application where nodes represent genes and the modules are represented by colors. Relationships are represented by grey lines that connect one gene to another. 4
1.2 A screenshot from GeneNet VR where a user is exploring the blood dataset from MIxT. 6

2.1 GeneNet VR. Example of the application running on a Oculus Quest. 8
2.2 Mapping of the Oculus Quest controllers for the different actions implemented in GeneNet VR: 1. Snap rotation. 2. Filter menu. 3. Scale environment. 4. Translate environment. 5. Pointer. 6. Select item in menu. 7. Oculus menu. 8. Teleport. Adapted figure from Oculus developer's page [21]. 10
2.3 Teleportation technique. The user can use the jystick from the right controller to teleport to a different spot. To choose the spot a parabolic arc will appear. 11
2.4 Translation of the network functionality. The user holds the translation button on the Oculus controller and moves the hand to the direction where he or she wants the network to translate. 12
2.5 Zooming in the network functionality. The user can hold the scaling buttons on the Oculus controller to make the network bigger or smaller. In this example if we strech our hands outside, the network will expand. 13
2.6 Diagram: steps for the creation of the network from the 2 CSV files. 16
2.7 Filtering menu in GeneNet VR. 18
2.8 Network morphing from the blood dataset to the biopsy dataset. 19
2.9 Architecture and design of GeneNet VR. 20
2.10 Network creator algorithm. 22
2.11 Algorithm for zooming in the network. 23
2.12 Algorithm for translating the network. 23
2.13 Algorithm for the selecton of the nodes in the scene. 24

2.14 Algorithm for the network morph. 25

3.1 Screenshot from the network view in the MIxT web application. 28
3.2 Hairball problem in the network view in MIxT. 29

4.1 Distribution of the number of edges in the blood dataset. The x-axis shows the number of edges and the y-axes shows the distribution in the nodes. 36
4.2 Bar graph showing a summary of the performance results for the 1% lowest average (7 frames with worst performance). . 38
4.3 Scatter plot showing the relation between the number of edges to render and the time that it takes to render for the blood dataset. 39
4.4 Profiling the selection of the node ARGLU1(1607), which has the highest number of edges. 40
4.5 Number of triangles in the scene when selecting the node ARGLU1, which has 1607 edges, the largest number in the blood dataset. 41
4.6 Performance of GeneNet VR when visualizing the blood dataset running on a machine and on the Oculus Quest. The x-axis represents the frame number (like a timeline). The y-axis represents the amount of time in milliseconds that a particular frame took to render. 42

5.1 Optimized virtual reality chemical space. Figure taken from [36]. 50
5.2 Screenshot from BioVR. Figure taken from [37]. 51
5.3 Screenshot from CellexalVR. Two users using CellexalVR at the same time. The head models were taken from NASA. Figure taken from [38]. 52
5.4 Screenshot from BigTop where a node is selected. Figure taken from [39]. 53

7.1 Design of a user interface for GeneNet VR with help of a graphic designer. 59

List of Tables

2.1 Fragment of the dataset with the categories and the genes belonging to each category from the biopsy sample. 15
2.2 Fragment of the dataset used to build the network relationships of the blood sample. 15

4.1 Elements of the network that have influence in the scalability. 32
4.2 Machine specification. 33
4.3 GPU specification. 33
4.4 Oculus Quest specifications. 33
4.5 Performance results in milliseconds when translating the network. 35
4.6 Performance results in milliseconds for the scale the network interaction. 35
4.7 Performance results in milliseconds for the select node interaction. 37
4.8 Information about the respondents. 44

1

Introduction

The cost of producing data has plummeted in recent years and many people are capitalizing on technological advancements to do so. In the field of genomics, the sequencing of the first human genome (2002) took around years and cost over $3 million to complete. Nowadays, it is possible to sequence hundreds of genomes in just a few days with a cost of around $1,000 each [1]. However, the vast amounts of data that are produced can result in a problem: data information overload. To solve this, new visualization tools are needed to help examine large volumes of data so that novel patterns in them can be found, which in turn will lead to new scientific discoveries.

In the field of system biology, there are usually network representations where the nodes or bioentities are connected to each other. These connections represent associations. Networks can increase dramatically in size and complexity and many visualization systems for biological networks lack scalability and the user interactions can be cumbersome. Virtual Reality (VR) has shown to have benefits when visualizing abstract information and offers rich interactivity [2]. Some specific challenges that we face when visualizing biological networks in VR are the following:

- Information overload because of the large number of nodes, edges, and information connected to those such as the weights and modules when visualizing large biological networks in Virtual Reality.

- Understanding the scalability limitations as for the number of nodes and

edges for the exploration of biological networks in VR.

- Performance requirements needed (72 FPS), so that the interactions are not cumbersome, using inexpensive VR equipment.

There are existing approaches for the visualization of large biological networks, but the have limitations. Many of them are 2-dimensional, some provide a 3-dimensional view and a limited number of them are compatible with Virtual Reality. These tools struggle with information overload problems or the "hairball" effect when the network becomes larger. Some tools like Cytoscape [3], NAViGaTOR [4] or BioLayoutExpress3D [5], help overcome these problems with specific hardware or libraries, other tools just trade off the interactivity in favour of showing large amounts of data [6].

We have implemented GeneNet VR, a virtual reality application for the visualization of large biological networks. We used two datasets from the MIxT project [7] that contain genetic information from patients with breast cancer. MIxT provides a 2-dimensional visualization tool to explore these datasets. However, it has some known visual and scalability problems. With GeneNet VR, we overcome these problems by providing the following solutions:

1. Visualization of the network in a three-dimensional immersive space and implementation of interactive and visual solutions to reduce information overload.

2. Design and implementation guidelines based on the evaluation results.

3. Implementation for the Oculus Quest, a cheap Virtual Reality headset.

We evaluated the performance of GeneNet VR and several of the interaction that are commonly used for network exploration. Another experiment was also carried out to evaluate the performance of the application on the Oculus Quest hardware. We concluded from the experiments that GeneNet VR performs well for the network sizes that we tested and that the interactions with the network achieved the required Frames Per Second (FPS) defined by Oculus. We also concluded that we can use inexpensive VR hardware to explore biological networks.

To evaluate the quality of GeneNet VR, we used a qualitative research where data was collected using semi-structured interviews with several scientific researchers. The feedback that we obtained was positive, hightlighting that the application is helpful for the visualization of biological networks and easy to learn even for novice VR users.

Thesis statement: *Virtual Reality is advantageous for the visualization of large biological networks and for rapid exploration of patterns in them using affordable hardware*.

1.1 Challenges and research problem

In fields such as biology, network visualization seems to be particularly helpful [8] [9]. There are many types of relationships that can be measured in a biological context, for example interactions between proteins or genetic interections when revealed by combinations of mutations. All these interactions and correlations can be easier to visualize as a network [10].

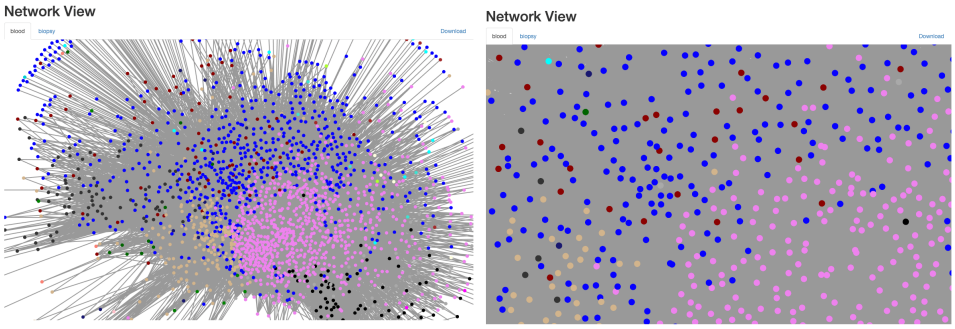
MIxT [11] is a web application for bioinformaticians that is used to identify genes and pathways in a primary tumor that are tightly linked to genes and pathways in the systemic response of a patient with breast cancer [7]. Among other tools, it offers a visualization tool for the biological networks where the nodes are genes and the edges between two nodes represent a statistically significant correlation in expression between them.

OmicsNet is another example of a visualization tool with data overload problems [12]. OmicsNet is a web application for creating different types of molecular interaction networks and visually exploring them in a three-dimensional space. However, the application also struggles with problems like edge occlusion and performance, making it hard to visualize the network when they become larger.

It can be hard to identify novel patterns when exploring large amounts of data. Some data structures, like the networks, also have the challenge of data interconnectivity. These data structures represent relationships and are composed of nodes and edges. Even though we have many tools like machine learning that help researchers automate and accelerate the identification of patterns, we still need expert human involvement expert to check and inspect these networks [13].

When exploring a network in MIxT, sometimes it can be difficult to find patterns because of the data overload issues. This problem happens particularly when there are many interconnecting nodes. In figure 1.1 we can see an example of the network visualization from MIxT. As we can see in Figure 1.1a, there are many interconnecting nodes; this problem is amplified when we zoom into the network as in Figure 1.1b.

There are additional challenges when exploring large biological networks in



(a) Network with several modules. **(b)** More focused image of the network.

Figure 1.1: Network view of the MIxT application where nodes represent genes and the modules are represented by colors. Relationships are represented by grey lines that connect one gene to another.

Virtual Reality. When we are in an immersive three-dimensional space, we have occlusion problems. This occurs, for example, when the nodes or edges that we have in front of us hide or obscure other nodes or edges that are behind them. One solution can be to show the network from another angle. We can do this by rotating and moving the network or by making it possible for the user to move to other parts in the virtual world. With reference to the issue of information overload, as highlighted earlier, this can be resolved by showing only the information that the user needs to visualize at any given time. This can be done by filtering the data so that we can focus on what we are interested in.

The interactivity in VR is usually done with the VR controllers, which simulate our hands in the virtual world. We can implement natural actions for the operator such as grabbing objects or items with their hands, which will feel intuitive. However, we can also implement other actions in the virtual world like using laser pointers to select something, using 2D menus inside the virtual world or teleporting the user to other parts of the virtual space. In GeneNet VR, we are dealing with abstract information and the amount of data can escalate quickly if we do not implement practical solutions. We must therefore maintain a proper balance between the amount of data being visualized, comfort and user-friendly interaction solutions and good performance.

1.2 Proposed solution and contribution

Virtual Reality offers new possibilities for visual inspection of large biological networks and for pattern identification within them. Even though VR is still a

field under exploration, it has been demonstrated that it helps scientists work more effectively in fields such as medicine [14] [15] [16], biology [17] [18] and neuroscience [19][20], to cite but a few examples. VR takes advantage of the way human beings perceive and analyse information naturally. Human beings have a great ability to discover patterns; however, they are biologically optimized to see the world and the patterns in an immersive visual 3-dimensional space. In addition, VR offers very rich interactive solutions.

Our solution, GeneNet VR, is a Virtual Reality tool that focuses on solving common problems for the visualization of large biological networks. To solve the information overload problem, GeneNet VR shows only the necessary information when exploring the network. The edges are shown for individual nodes when the user selects them. It is also possible to move around the virtual space and the user can scale, move the network for a better angle and to solve the occlusion problem. In addition, a filtering menu was implemented to filter the information that the user wants to see. We implemented a case study where we visualize networks from biological datasets that are used in another scientific project. This helped us understand the limitations for the visualization of these type of networks. We also implemented a feature that enables the user visualize two networks simultaneously, which is useful for datasets like the ones that we used.

We have evaluated the performance, scalability and the quality of GeneNet VR. The pattern exploration in these datasets is an important process that cannot be interrupted by low FPS. We measured the time frame in the experiments running GeneNet VR in a machine and obtaining an average of 7-8 milliseconds, which is under the 13.9 milliseconds limit that corresponds to 72 FPS. We also evaluated the performance on the Occulus Quest headset, reaching 72 FPS for most of the frames and making affordable all-in-one VR headsets a good option to visualize these datasets.

To evaluate the quality of our project, we used a qualitative research approach during which we conducted purposive sampling with semi-structured interviews with biologists, computer scientists and pharmacoepidemiologists from UiT. We learned from the interviews that GeneNet VR is a good solution for the visualization of large networks. The performance and the interactions were also very smooth according to the respondents, and the application was easy to use, even for novice users in VR. We also obtained interesting feedback for future improvements and also for the visualization of similar networks like drug and social networks.

This project has contributed by tracing out some of the important requirements that are needed for the visualization of large biological networks in VR. GeneNet VR gives visualization experts innovative and engaging tools to

explore biological data. In Figure 1.2 we can see an example of GeneNet VR, where a user explores the blood dataset from MIxT. The node TMED7 is selected and its edges are shown. The user is also using a UI menu to filter the nodes. We also made a video where we show the several interactions implemented for GeneNet VR. The video can be viewed on YouTube [1].

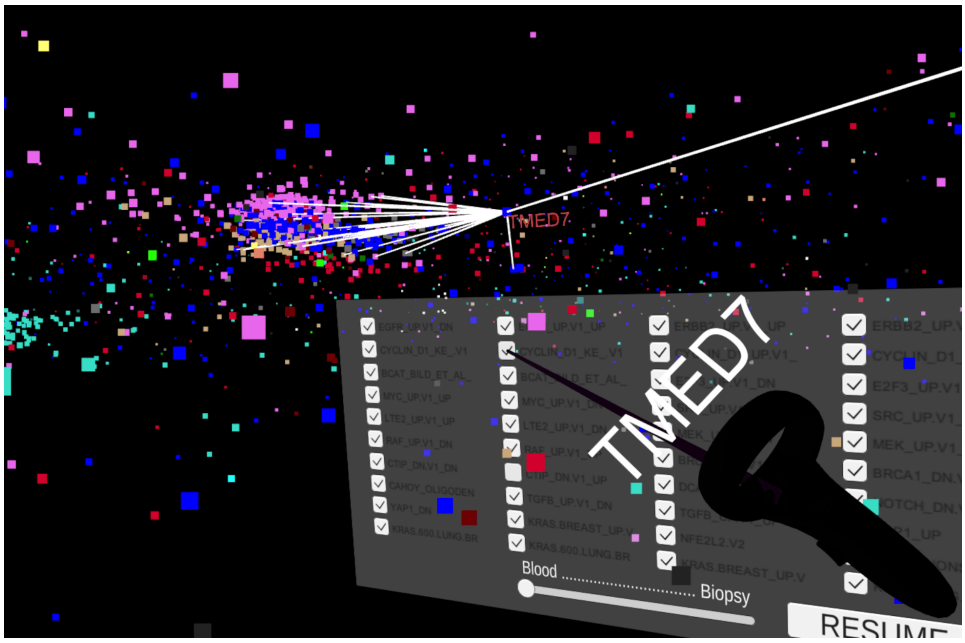


Figure 1.2: A screenshot from GeneNet VR where a user is exploring the blood dataset from MIxT.

1.3 Outline

We have structured the thesis in the following chapters: Chapter 2 describes how GeneNet VR was implemented, the architecture, design and the Virtual Reality techniques that were used. Chapter 3 focuses on explaining the visualization of MIxT in Virtual Reality and the datasets that we are using. In Chapter 4 we describe the experiments we carried out and our conclusions which we used in order to evaluate GeneNet VR. Chapter 5 describes some related projects found in scientific literature and we draw comparisons with GeneNet VR. In Chapter 6 we explain the conclusions from the project. In Chapter 7 we describe the future development ideas that we have for GeneNet VR.

1. https://www.youtube.com/watch?v=_Iqa3yizYZ4

2

GeneNet VR

GeneNet VR is a virtual reality application for the interactive visualization of gene networks in a 3D space. The network is represented using nodes and edges between them. In order to explore and visualize the data in GeneNet VR, the user can for example walk around the 3D environment, zoom in the network, translate it to other places, filter the nodes using a user interface, morph the network from one dataset to another and also obtain detailed information about the nodes.

GeneNet VR loads the data from local files with the information about the nodes and relationships. Then the network is built using the data and clustered using an algorithm. Finally, the user can explore it and interact with it using the VR headset and controllers.

We implemented GeneNet VR in Unity, a cross-platform game engine. This software is used for a wide range of applications, especially for the development of videogames in 3D and 2D, VR applications and engineering solutions. We used C# as the main programming language to develop the application in Unity. We also used VRTK, a VR toolkit to build VR solutions in Unity. As for the VR hardware, we used an Oculus Quest headset. This type of headset is an all-in-one HMD, which means that it doesn't need to be connected to a PC to run an application, it has its own hardware to run the applications although this can be more limited than the hardware from a PC. Also, during the development process I used a cable and Oculus Link, a software to connect Oculus Quest to the PC, to run the application and test it directly from Unity

on the Oculus Quest, without having to load it to the headset. This was from great help during the development process.

We have chosen two datasets from MIxT to develop GeneNet VR. MIxT is a web application that is used for exploring and comparing bioinformatic data [11] [7]; and the data visualization is an important part of the process. The datasets used here contain genetic information about a woman with breast cancer. There are in total 2 tissues; the first one is from a blood sample and the second one is from tumor tissue. In Figure 2.1 we can see an example of GeneNet VR running using the blood dataset from MIxT. We will now go in deep with how we implemented GeneNet VR.

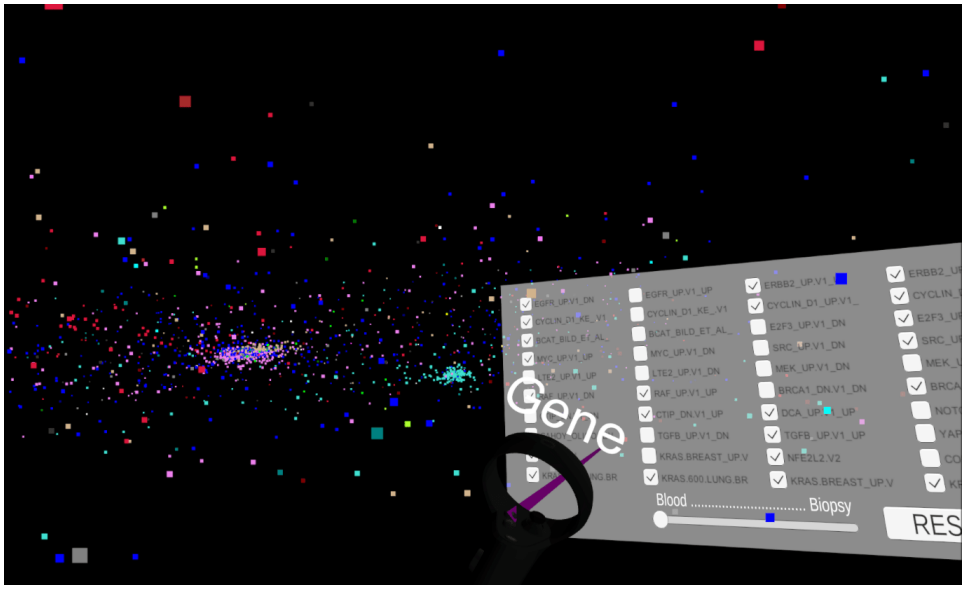


Figure 2.1: GeneNet VR. Example of the application running on a Oculus Quest.

2.1 Interaction with the network

Virtual reality headsets offer a rich inmmersive experience. It's not only about immersing the user into a 3D environment, but also giving the user the possibility to interact with the environment itself. This makes it possible to build complex VR applications where the user can do almost anything in a virtual world. Some examples of what it is possible to do in virtual reality is for example moving around, grabbing objects, interact with the environment using your hands or virtual tools, like pushing a button, 2D interfaces and menus, etc. In this section we will explain the techniques that we have implemented to visualize and interact with the network and make the most of VR.

The interaction and the visualization also depends on the VR technology used. We use Oculus Quest in this project, an all-in-one VR headset that doesn't need a PC nor wires to run the applications. Apart from the headset, it comes with 2 controllers; one for each hand. These controllers have inputs as buttons, thumbsticks and triggers that can be used to activate actions in the VR application. We have used some of these inputs available in the controllers in GeneNet VR and mapped them to different actions that allow the user interact with the network and the environment.

In Figure 2.2 we can see which actions correspond to each input from the controllers. We will briefly explain now what these actions consist of: 1. Snap rotation: It allows the user to instantly rotate to the right or to the left 45°; 2. Filter menu: The user can filter the nodes of the network according to a filtering algorithm used in GeneNet VR; 3. Translate network: The network can be translated or moved to other positions in the scene; 4. Scale network: The network can be scaled or "zoomed"; 5. Select node: The user can select a node in order to get more information about it; 6. Select item in menu: It allows the user interact with the menu, for instance to filter the nodes by enabling or disabling the checkboxes from the filtering menu; 7. Oculus menu: It opens the menu from oculus and pauses GeneNet VR; 8. Teleport: It teleports the user to another position on the floor of the VR scene.

As for the use of the Oculus Quest HMD (Head Mounted Display), this is placed in the head and it has a belt that is used to adjust the headset to the head. This will help the user feel more comfortable while wearing the HMD. Another important aspect that we have taken into account in GeneNet VR is that the user can use the application and explore the network by sitting on a chair. This is possible thanks to the locomotion techniques implemented that allows the user to move around with the controllers. We will go into more detail later in this chapter about this.

We will explain now in the following subsections the different interaction techniques that we have used and also what benefits they bring for the visualization and interaction of the network.

2.1.1 Locomotion

Locomotion is one of the most important ways of interaction in virtual reality experiences. It can be defined as a self-proppelled movement in the virtual world. Even though moving around is not the main goal in most of VR applications, it is an important aspect for the user's perspective in order to move the user's viewpoint in the virtual world and navigate around it.

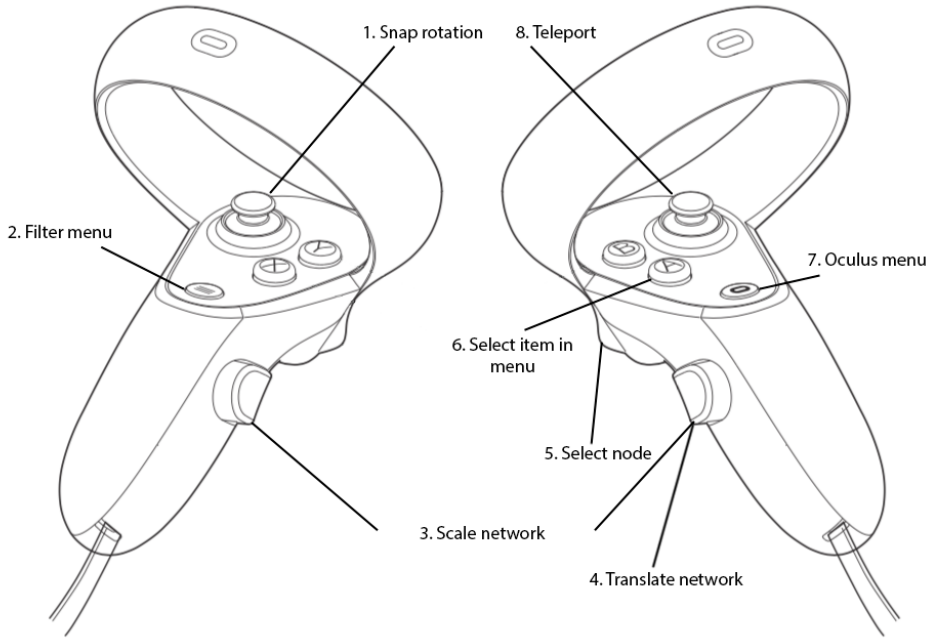


Figure 2.2: Mapping of the Oculus Quest controllers for the different actions implemented in GeneNet VR: 1. Snap rotation. 2. Filter menu. 3. Scale environment. 4. Translate environment. 5. Pointer. 6. Select item in menu. 7. Oculus menu. 8. Teleport. Adapted figure from Oculus developer's page [21].

Locomotion can have an strong influence in the user's experience. A poorly designed locomotion technique can reduce the user's immersion and even introduce motion sickness, which is related to the movement that the technique produces. HMDs like Oculus Quest allow the users to control the position and the orientation of the viewpoint by moving their heads and walking; however, large virtual environments such as GeneNet VR need a big physical tracked area, which cannot be covered by just walking around. It is for this reason that we need to use a locomotion technique that makes it possible to move around without having to walk around in the physical world [22]. In addition, as research has shown, when the user is stationary both in the virtual and real world, the motion sickness produced by VR is less likely [23].

The locomotion technique that we use in GeneNet VR is called teleportation. It consists in choosing a spot on the floor were we want to teleport to. To do this the user has to move forward the thumbstick from the right controller (see "8. Teleport" from Figure 2.2). Furthermore, it is possible to choose which direction the user will face once the teleportation is completed. To do this we just need to rotate the same thumbstick to the desired direction. Once the

user releases the thumbstick, a black flash will be followed by the new position in the space. This black flash is very important when implementing some of the locomotion techniques because it prevents from producing motion sickness and disorientation. Without the black flash, the transition to the new position would be too abrupt and it may disorient the user.

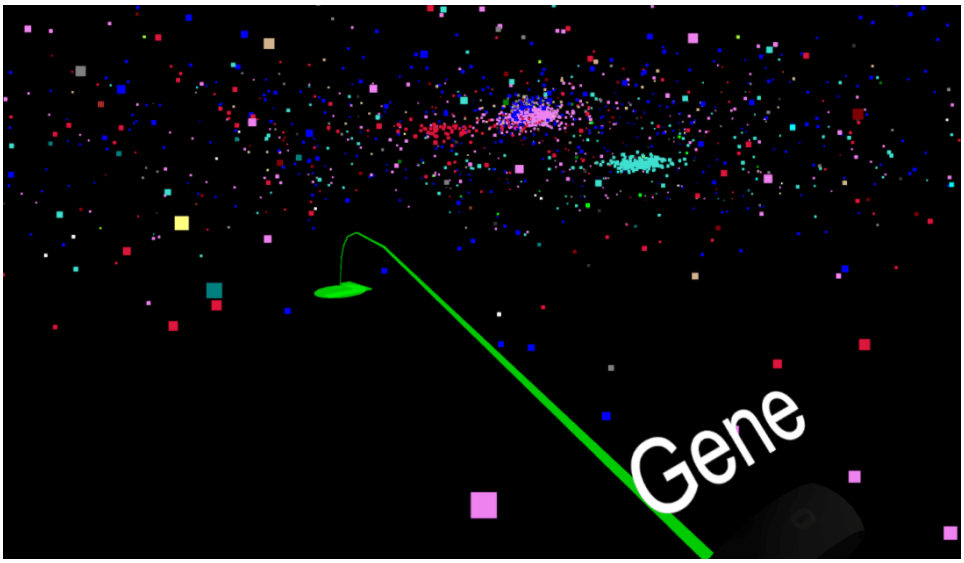


Figure 2.3: Teleportation technique. The user can use the jystick from the right controller to teleport to a different spot. To choose the spot a parabolic arc will appear.

In Figure 2.3 we can see an example of how the teleportation technique is used in GeneNet VR. A parabolic arc is created in the 3D space with a circle representing the spot where we are goint to teleport to. It can be seen as if we are throwing an object to the spot where we want to teleport to. The green circle includes also an arrow, indicating the direction that we will face once we are teleported.

In addition to the teleportation, it is also possible to rotate to the left or to the right with the Oculus controllers so that the user doesn't have to rotate the head to look around in the scene. This action is triggered using the thumbstick on the left hand (See 1. Snap rotation in Figure 2.2). By moving the thumbstick to the left side, the camera will rotate 45° to the left side, and 45° to the right side if the user moves it to the right side. A black transition is also used in this case before the rotation happens to avoid motion sickness, for the same reason as in the teleportation technique.

2.1.2 Translation of the network

By teleporting to different places in the environment, we allow the user visualize the network from different perspectives; however, it is also interesting to be able to move the network and specially move it in a precise way, so that the user has more control over what it is being visualized. The user might for instance be able to see the network or a specific node or cluster from above or also from below. To do this we have implemented a functionality to translate the network in the 3D space.

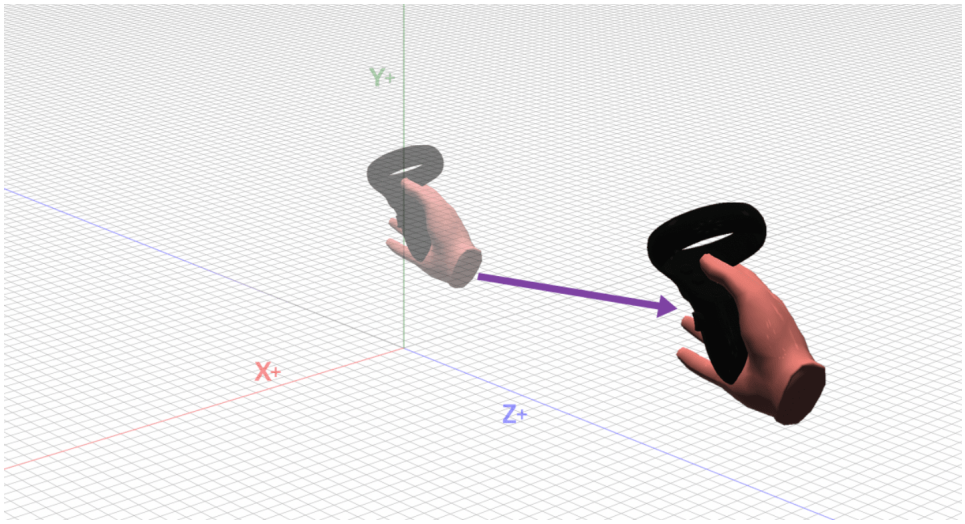


Figure 2.4: Translation of the network functionality. The user holds the translation button on the Oculus controller and moves the hand to the direction where he or she wants the network to translate.

To translate the network in GeneNet VR, the user needs to press on the hand trigger from the right controller (see "3. Translate network" in Figure 2.2). Then the user needs to keep holding this trigger down and move the hand to the direction to which we want the network to move to (see Figure 2.4). This intuitive approach feels like we are just pulling from a rope tied to the network and we just move it to the direction we want.

2.1.3 Zooming in the network

When exploring a big network with hundreds of nodes and several clusters, sometimes the information can be too crowded. In our example dataset that we use in GeneNet VR, there are some clusters of nodes that have too many nodes close to each other and it gets very hard to visualize them properly. A way to cope with this problem is for instance by "zooming" in the part of the network

that we want to explore better. We implement then a scaling functionality that makes the network bigger or smaller.

The way we implemented the zooming functionality in GeneNet VR is by using the hand triggers with the name "4. Scale network" (see the reference in Figure 2.2). In the first place the user needs to press and hold these triggers from both controllers, and then we need to expand or contract the arms, as if we were stretching out or contracting the network itself. This is also an intuitive action to do since the user might think that we are actually stretching the network with the hands.

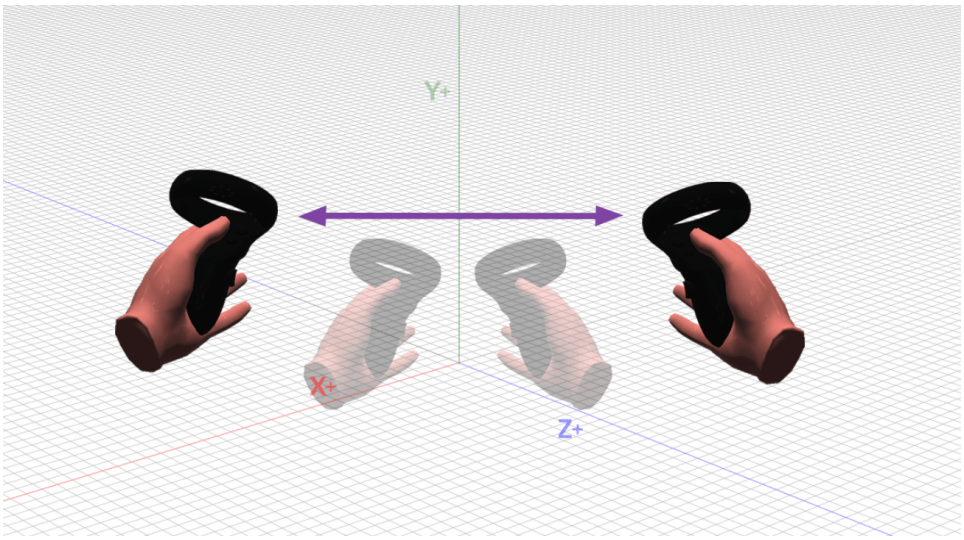


Figure 2.5: Zooming in the network functionality. The user can hold the scaling buttons on the Oculus controller to make the network bigger or smaller. In this example if we strech our hands outside, the network will expand.

In Figure 2.5 there is a visual example of how the zooming works using the Oculus controllers. In this example the user is stretching the hands out in order to make the network bigger. The user starts in an initial position, then holds the zooming triggers from both controllers and then moves the hands out. If we wanted to make the network smaller we would do the opposite action, by contracting the hands to the inside.

2.1.4 Interaction with the nodes

GeneNet VR provides also information about the data that is being displayed. The user can interact with the nodes of the network to obtain information about each of them. In our example, the nodes represent genes and the user might be interested in knowing which gene name corresponds to a specific node.

The action that we need to do to obtain the name of the gene is to get close with the right controller to the node that we are interested in and press the "5. Select node" index trigger on the right controller (see Figure 2.2). When we press this trigger, we can select a node from the network. By selecting a node, we will get the name of that gene node that will be displayed in a rendered text, and we will also visualize the edges from this node to other nodes. The node is selected with an algorithm that searches for the node closest to our right controller.

2.1.5 Node relationships

Finally, our dataset has information about the relationships between the nodes. GeneNet VR is implemented to show also this information. Because there can be too many relationships in the dataset, we don't show them all at the same time. Therefore, we can only see those of the node that the user has selected. The way that these relationships are represented is with lines between the nodes.

2.2 Scalable network in Unity and data structures

GeneNet VR uses files from an external source with data that will be used to be the network. The first file contains the information about the nodes and what category the node belongs to; The second one has information about the relationships between each of the nodes. As for the content of the files look like, in Table 2.1 and Table 2.2 we show an extract from them. Originally the files are in CSV format. CSV [24] stands for Comma-Separated Values where each record is located on a separate line within the file, delimited by a line break. In addition, each record can contain one or more fields, separated by commas.

For our example we represent the extract from the CSV files using tables, which are more illustrative. Table 2.1 contains an extract from the file with information about the genes and the categories to which each gene belongs to. Here each row is a category and as we can see, the second cell contains all the gene names for that specific category. These categories are named by colors and these color names will be used by GeneNet VR to color each node from the network. As for the second table, Table 2.2, this one shows an extract with the information about the relationships between the genes. This file can be very large since each row in the CSV file corresponds to a relationship between two

genes and one gene can be related with multiple genes. For instance one of the CSV files that GeneNet VR uses to build the relationships contains almost 90k lines.

category	genes
brown	ARHGAP30 FERMT3 ARHGAP25 CD53 PLEK IRF8 DOCK2
cyan	SAFB MOB3A RAB35 ABR ASCC2 CDC37 ANKFY1 GLTSCR1
darkgrey	RAB40C ZNF213 ZNF263 PIGQ RHBDF1 RAB11FIP3
darkorange	TCEB1 MRPL13 ENY2 MTERF3 UBE2W WDYHV1

Table 2.1: Fragment of the dataset with the categories and the genes belonging to each category from the biopsy sample.

source	target	weight	id
AAMP	ARGLU1	0.102486209330144	AAMP-ARGLU1
ACADM	FOXN2	0.107506881676173	ACADM-FOXN2
ACADM	MBNL1	0.12269622045714	ACADM-MBNL1
ACADM	PPM1B	0.103496640767895	ACADM-PPM1B

Table 2.2: Fragment of the dataset used to build the network relationships of the blood sample.

The following diagram shown in Figure 2.6 schematizes the steps that we follow to build the network in Unity. We start with the 2 CSV files described before, containing the data about the nodes and the relationships. We process these CSV files in order to store the data in data structures in GeneNet VR. During the process of storing this data we also apply a clustering algorithm that will set the correct position for each node in the network. After doing this we can easily access the information about the nodes, their position, color and to which nodes they are related to in order to draw the edges. Finally, the network is created using a particle system as we will explain later.

Now that we know how sources of information to build the network in GeneNet VR, let's take a look at how the network itself is represented in Unity and what algorithms and data structures we use for that. These will have an impact in the scalability of the network, and therefore it's important to choose a good solution. We have three elements from the network that have more weight in its scalability: the nodes, the edges and the cluster algorithm used. We are going to explain each of them in more detail now.

The nodes in GeneNet VR are represented using a particle system. In Unity a particle system [25] is defined an array of particle objects. Each particle is a defined structure in Unity that contains properties like the life duration of the particle, start color, start size or position in the 3D space. Particle systems in Unity are very useful to render some special effects like fire, steam, fireworks

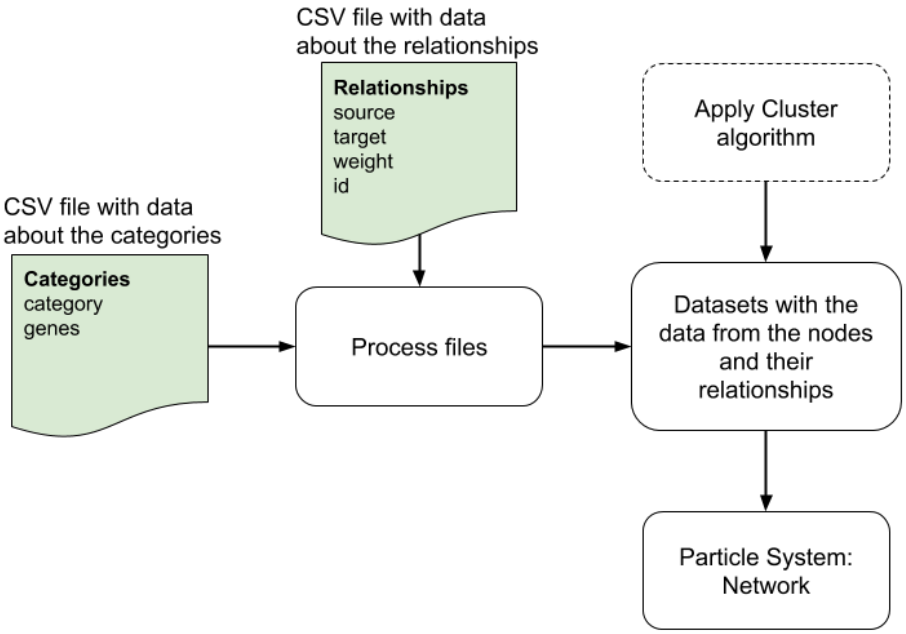


Figure 2.6: Diagram: steps for the creation of the network from the 2 CSV files.

or projectiles. They are also very powerful because they give plenty of control to the developer over the particles. In GeneNet VR we take advantage of this, allowing us to structure the network in the way we want. Usually particles have a lifetime, which means that for instance they can start in a position with a particular colour and finish or disappear after a few seconds in a different position and colour. In GeneNet VR, the particles are static and have a very long lifetime, giving the perception that the network is a rigid structure. As for the way the particles are rendered, a 2-dimensional square is shown in the scene for each particle or node. Finally, in order to store the information of each node or particle we have a dictionary object in GeneNet VR that looks like this:

```
private Dictionary<string, ParticleSystem.Particle> particles;
```

A dictionary in C# is a data structure that contains a set of keys and each key has a single associated value. In our case the key corresponds to the name of the node, the name of the genes in this case, and the value is a particle object.

The edges between the nodes are represented with 2-dimensional lines in GeneNet VR. A line in Unity is created with a Line Render component [26]. These lines are very flexible and can be used to draw anything from a straight

line to a spiral. They also have properties like color, texture mode, possibility to have different widths along the line, etc. In our case we want to render straight lines, so we need to know the start and the end points where the line will be rendered. This information is taken from the CSV file with the edges information. To store this information, we also use a Dictionary, where the key is the name of a node and the value is a list with all the nodes to which this node is connected to. This looks like this in C#:

```
private Dictionary<string, List<string>> edges;
```

Showing all the edges at the same time in GeneNet VR would make it very hard to visualize the network. For this reason we show only the edges of the node that the user has selected. Also, in GeneNet VR the edges are shown dynamically, meaning that they are created every time the user selects a node. When the user selects a different node, the current edges are removed and a new set of edges are rendered for the new node. This process can be a bit CPU-consuming in Unity. Every time an edge has to be rendered an edge object is instantiated with the CreateInstance method from Unity. The edge object in the scene from what is called a prefab in Unity. A prefab [27] is basically a reusable asset, which in our cause is the line with some defines properties like the width and the colour.

The algorithm used to cluster the nodes in the network is another important aspect that can influence in the scalability. In GeneNet VR we use a linear algorithm that clusters the nodes in the 3d space depending on the module where they belong too. In this way the user can visualize each module as single clusters with a distinct colour per cluster.

2.3 Other features of GeneNet VR

GeneNet VR provides some features that help in the process of visualization and interaction with the network. They have a complementary purpose, and they don't influence much in the scalability of the system.

2.3.1 Filtering information in the network

Another feature that GeneNet VR uses to improve the visualization of networks is a filtering menu. When we have huge amounts of data in large networks, it is sometimes necessary to show less or more data. By filtering the nodes we can visualize only the part that we are interested in.

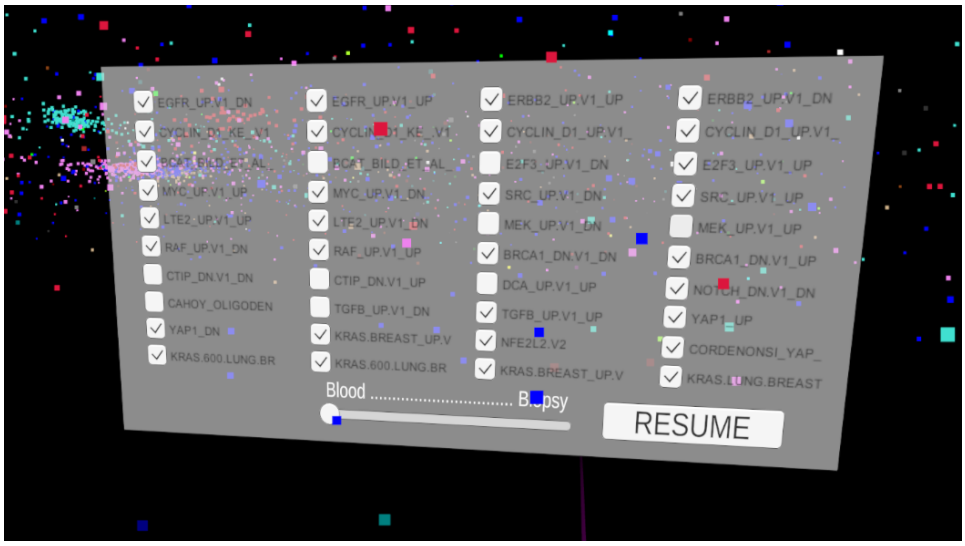


Figure 2.7: Filtering menu in GeneNet VR.

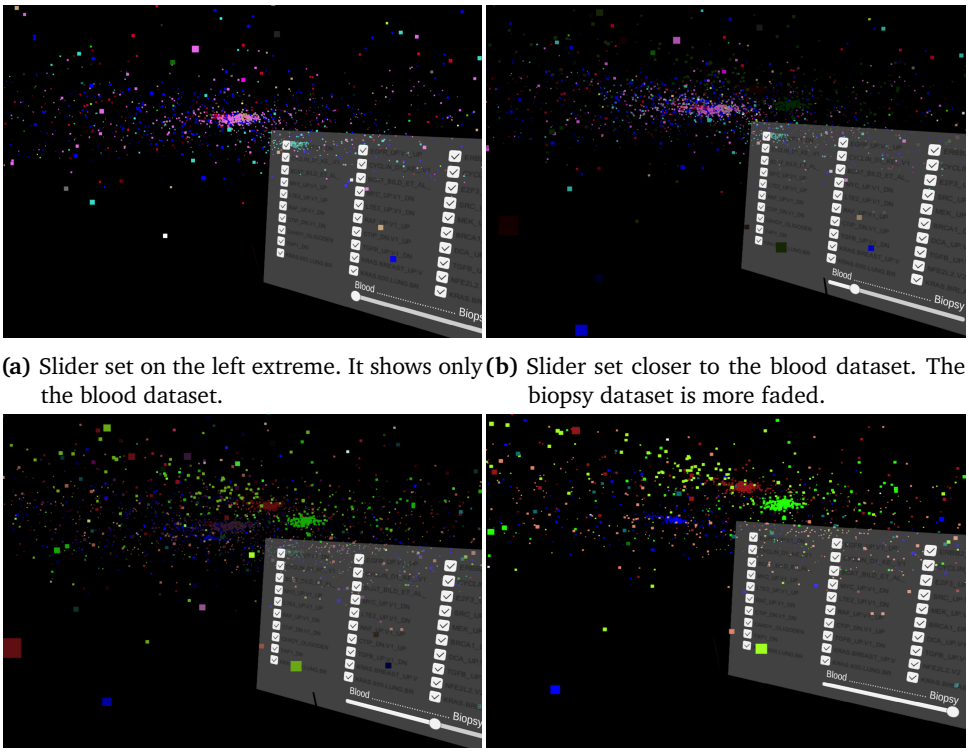
We have built a 2 dimensional menu in Unity, see Figure 2.7, to filter the data in our example network. We use checkboxes for the filtering. From a starting point, all the boxes are checked, and if the user wants to hide a part from the visualization it is done by unchecking the box. To show the filtering menu we need to press on the menu button from the left controller, see the "2. Filter menu" in Figure 2.2. The to check or uncheck the boxes we need to use the A button from the right controller, named "6. Select item", see Figure 2.2.

2.3.2 Network morphing

GeneNet VR has also the possibility to morph from one network to another. This can be done in the filtering menu by pressing the menu button from the left controller and there we can see a slider as in Figure 2.7 which we can move to the right or to the left in order to morph the network. In Figure 2.8 we can see an example of how the network morphs, showing 4 states of the morphing process. In a) the network is showing the blood network state and on d) the biopsy state. In b), the slider is moved slightly to the right, but closer to the blood state. We can see that in this state, the biopsy network is starting to show. Also, some nodes from the blood network are starting to move to relocate to their position in the biopsy network. In c) the slider is set closer to the biopsy state (to the right extreme) and we can appreciate more clearly the biopsy dataset, but the blood one is less visible.

This morphing tool help us visualize two datasets at the same time and compare them. The slider has values from 0 to 10. When the value is set to 0 we visualize

the blood dataset, and when it is set to 10, we visualize the biopsy dataset. For the values that are neither 0 nor 10, we can see both datasets at the same time. Depending on if the value is closer to 0 or 10, the nodes from one dataset or the other will be more visible. In addition, the position of the nodes that are found in both datasets, is interpolated and there for we can see how this nodes move from one dataset to the other one by using the slider. Something that this tool doesn't allow us to do, is the selection of nodes and edges to render. We can only select the nodes if we are in either the blood or the biopsy state.



(a) Slider set on the left extreme. It shows only the blood dataset.

(b) Slider set closer to the blood dataset. The biopsy dataset is more faded.

(c) Slider set closer to the biopsy dataset. The blood dataset is more faded.

(d) Slider set on the right extreme. It shows only the biopsy dataset.

Figure 2.8: Network morphing from the blood dataset to the biopsy dataset.

2.4 Implementation details

Unity (version 2018.4.10f1 [28]) is the software that was used to build the system. It is a multi-platform game engine. It is known to be easy to use and for having a big community of creators and asset designers [29]. Even though it is intuitive to use, it also has low-level access for developers. As for Virtual Reality, Unity has been up-to-date with the new VR technologies thanks to professionals and amateurs in this area who have built an integration for Unity.

In our case, our device is an Oculus Quest, and for this reason we use the Oculus integration for Unity [30]. In addition, we have used VRTK, a collection of scripts and assets that help build VR solutions [31]. Finally, the programming language used in Unity to implement the system is C#.

2.5 Architecture and design

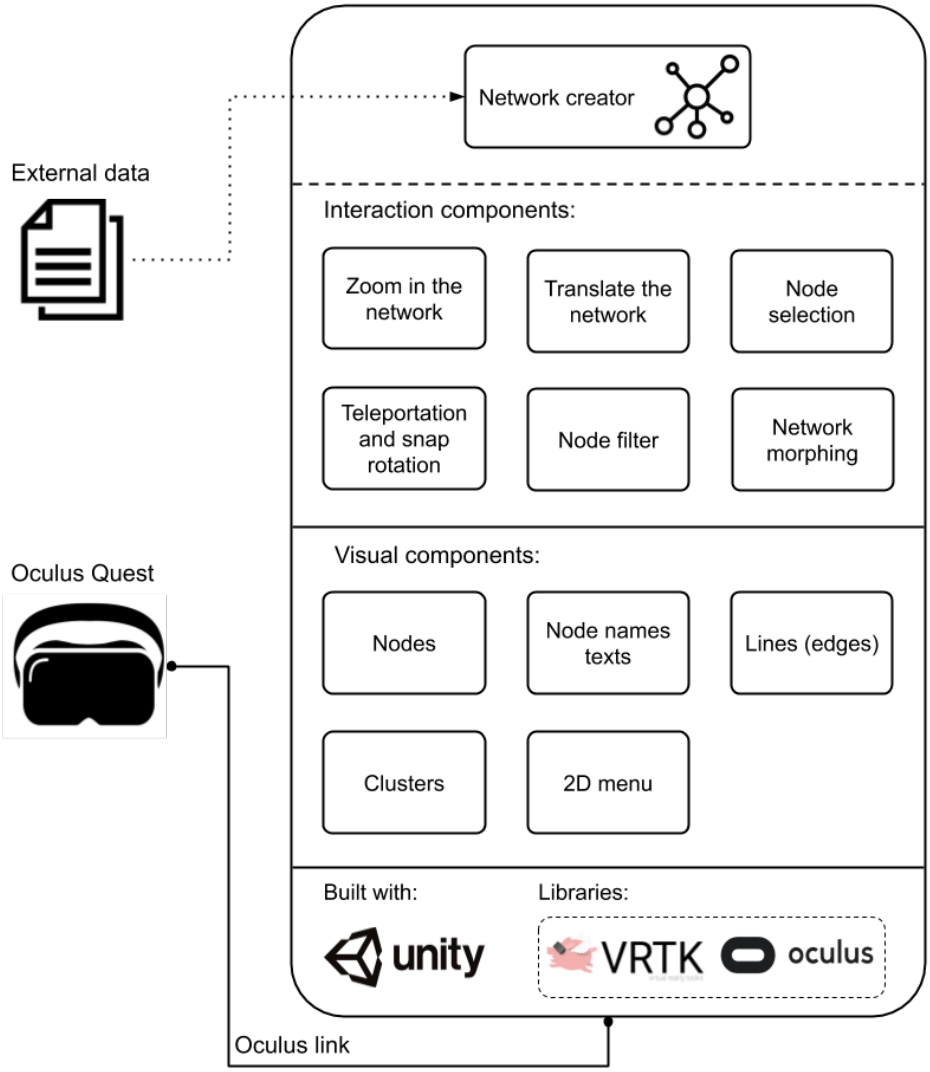


Figure 2.9: Architecture and design of GeneNet VR.

GeneNet VR is a VR application built in Unity. For the implementation of the different visualization and interaction components I programmed some C#

scripts, used also solutions that are native in Unity like the particle systems and made use of the VRTK library and the Oculus library for Unity.

In Figure 2.9 we have an overview of the architecture of GeneNet VR. The big box represents the Unity application and it contains all the components and functionalities that I have developed for the project. We can see that the big box is divided in 4 regions. The first one, starting from the top, is the network creator component that uses external data in order to build the network. In the second region we have the different interaction components that are available for the user to interact with the network and the environment. The third region contains the visual components that help the user visualize the data. Finally, the last region contains the technologies and libraries that I have used to build the application. We can also see the Oculus Quest headset represented down on the left. Here the user can visualize the network and use the controllers to interact. As we can see in the figure, the Oculus Quest can be connected to the PC using an Oculus Link, which is basically a high-quality USB 3 C to C or USB A to C cable with proven performance [32]. This allows the user run GeneNet VR on the PC. Another possibility is also load GeneNet VR in the Oculus Quest and run it in the hardware of the headset without any cable or PC. We are going to explain now in more detail how each of the interaction components of GeneNet VR were implemented. I will use some flowcharts to explain some algorithms. In this section we will also mention some actions that are triggered using the Oculus controllers. These are specified in Figure 2.2.

The network creator (see Figure 2.10) initializes and builds the networks using the data files that were previously stored in the application's directory. It processes the data from the CSV files and stores the information in hash maps that can be later be used by the interaction components to transform or read the data of the networks like the node positions or colors. During this process of building the network, we apply a cluster algorithm as well. This algorithm consists of a loop of 10 iterations where for each iteration we go through each of the relationships from the relationship file. For each relationship we update the position of the nodes so that the ones that are connected are closer to each other in the space, resulting into clusters of nodes depending on how related they are.

For the *zoom in the network* component I wrote a C# script where I use the Oculus integration to communicate with the Oculus Quest controllers. The algorithm is run every time the user enables the action for zooming (see Figure 2.11 for the algorithm). What the script does is to find out if the user is stretching or contracting the arms. For this, when the user triggers the action, the system stores the current position of the left and right controllers in the 3D space and calculates the distance between these 2 points. This position is called initial position. Until the user releases the triggers, the system calculates for every

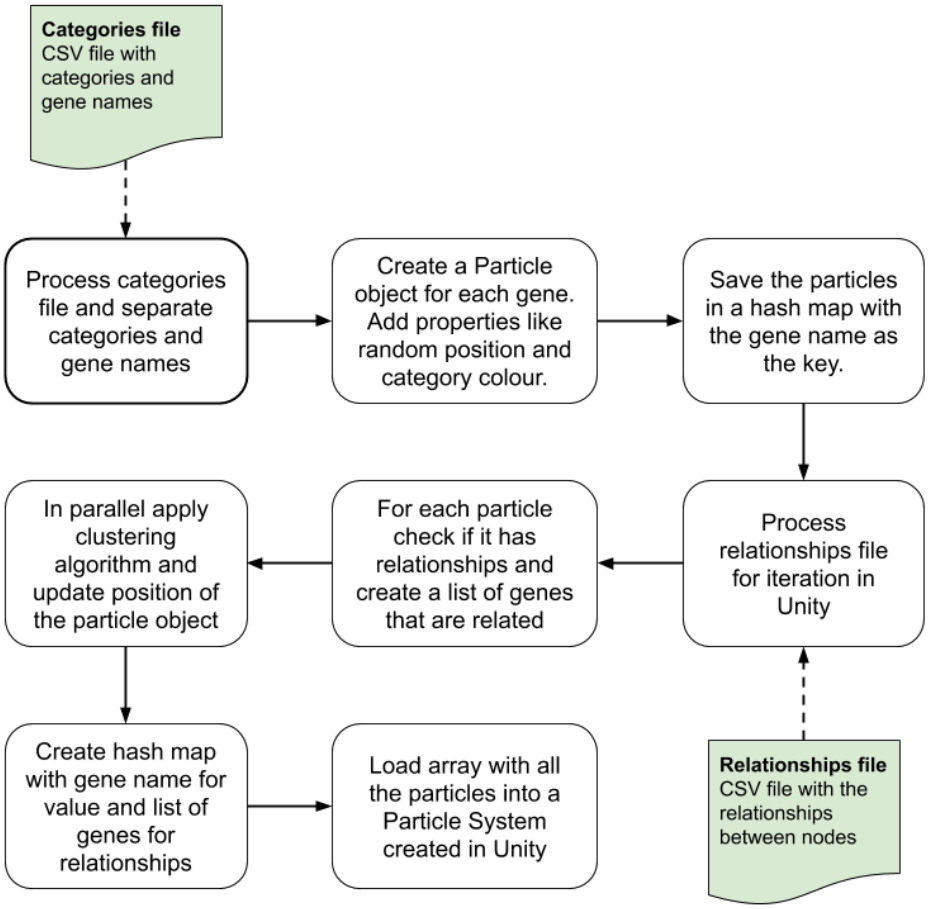


Figure 2.10: Network creator algorithm.

frame the current distance between the two controllers and compares it with the initial distance that was stored right before the user triggered the action. If the new distance is smaller than the initial one, the network will shrink; if it is bigger, the network will grow up.

The *translate the network* component consists is implemented in C# as well (see Figure 2.12). Here we do something similar to the *zoom in the network* component. When the action for the translation of the network is triggered, the position of the right controller is stored as the initial position. Then, while the user is holding the trigger of the right controller, the current position for the controller is calculated for every frame. A vector is calculated like (current_position - initial_position) and normalized. With this we obtained the direction of the vector where the user is trying to move the network to. We add up that vector with a constant to update the position of the network in the 3D space.

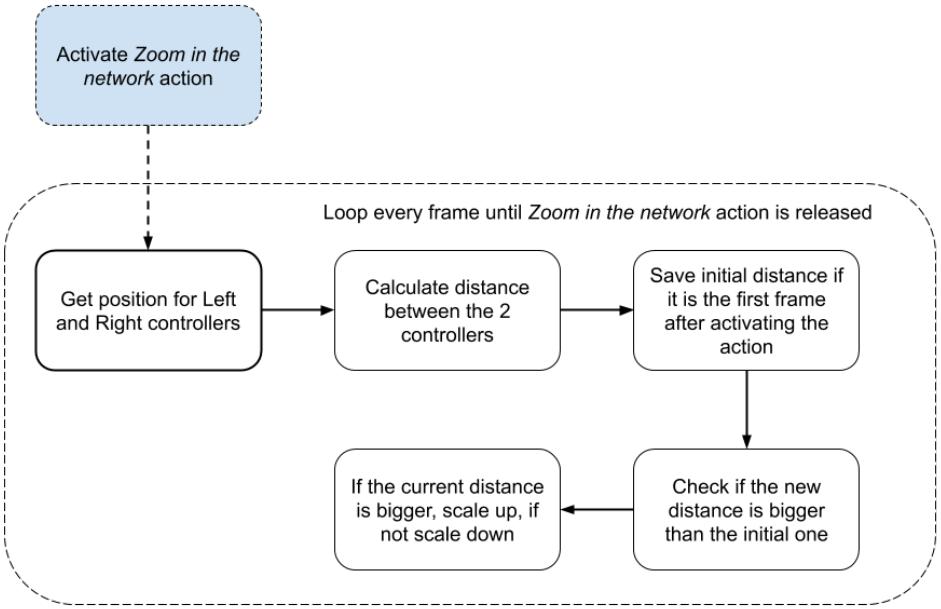


Figure 2.11: Algorithm for zooming in the network.

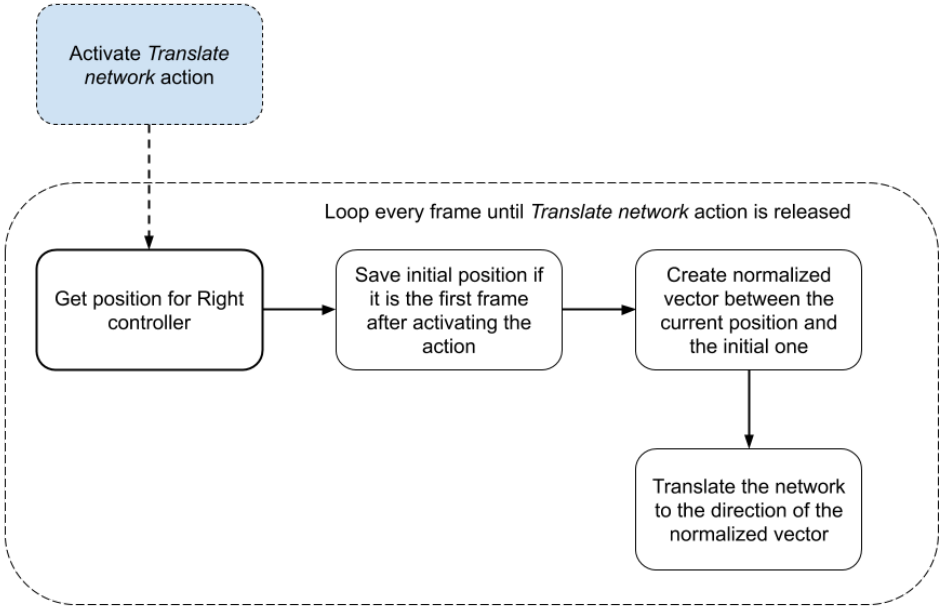


Figure 2.12: Algorithm for translating the network.

For the *select node* component, when the user triggers the *select node* action, an algorithm is used to calculate which node is the one that the user is trying to point at. In Figure 2.13, we can see a flowchart of the algorithm that we

implemented. I used C# to code a script for this functionality. A laser pointer comes out from the right controller when the user triggers the action. In the algorithm we make use of this laser information which consists of a vector. In order to select the node that we are pointing at, we calculate for each node in the network a vector product composed by the laser vector and a vector that goes from the right controller position to each node position. The result is another vector where we extract its magnitude. The magnitude that is smaller will correspond with the node that is closer to the laser pointer. We will select this node with the smaller magnitude value. When a node is selected, the algorithm also draws the lines corresponding to the relationships of this node in the scene. If there were any lines before the new node is chosen, these are removed.

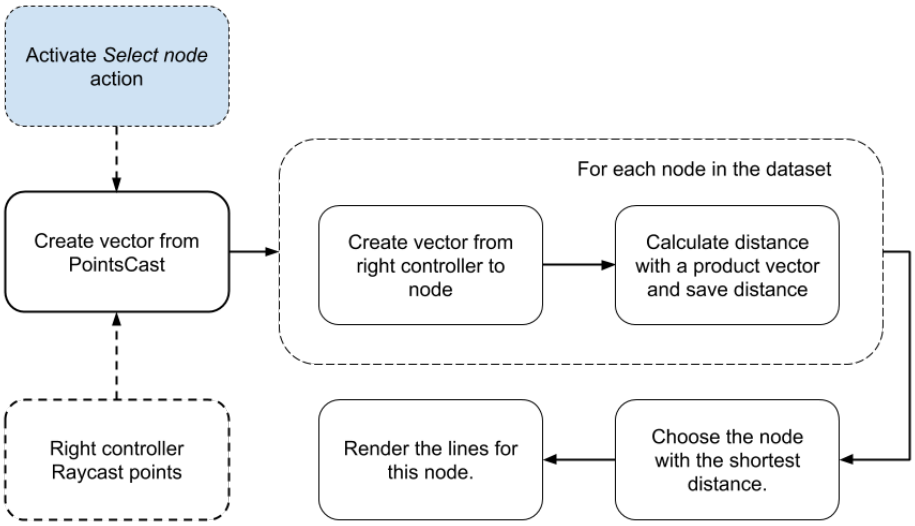


Figure 2.13: Algorithm for the selecton of the nodes in the scene.

The *network morph component* (see Figure 2.14) consists of a UI slider element that is in the menu and a method where we pass the value of the slider as a parameter. The values from the slider range from 0 to 10 and as the algorithm shows, value 0 will show the blood network in the scene and value 10 will show the biopsy one. When the value is greater than 0 or lower than 10, we interpolate the color and the position of the nodes. For the colour interpolation, we use a linear interpolation function from Unity called Color.Lerp. The position interpolation is only applied to the nodes that are in both the blood and the biopsy dataset. For this interpolation, we create a vector that goes from the node in the blood dataset to the node in the biopsy one. We divide this vector into 10 and multiply it for the slider value. This will be the new position for the node.

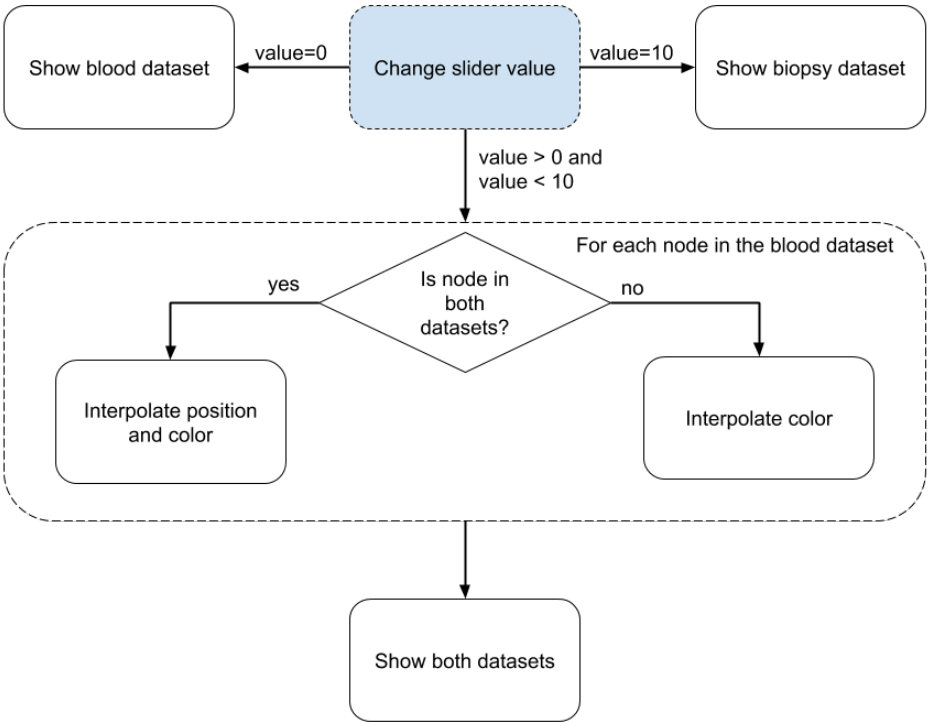


Figure 2.14: Algorithm for the network morph.

For the *node filter component*, a UI interface is used in Unity. This interface contains several checkboxes that are switched on by default. In addition, each checkbox has attached a function that is run every time the user switches it off or on. This function receives as a parameter a string value that is different from each checkbox. When the user switches the checkbox, the algorithm looks into a hash map that looks like this:

```
private Dictionary<string, string[]> oncoGroups;
```

We use the string variable that is passed to the function to look up the oncoGroups hash map, where each key corresponds to the name of each checkbox. We obtained a list of nodes that we need to turn on or off from the network.

Finally, for the *Teleportation and snap rotation component* we have used some scripts and prefabs that come with the VRTK library. I used as a reference some techniques to develop these locomotion solutions from a course on Unity[1].

1. https://learn.unity.com/course/oculus-vr

3

MIxT

The Matched Interaction Across Tissues (MIxT) is a system developed by UiT and Concordia University for exploring and comparing transcriptional profiles from two or more matched tissues across individuals [11]. This system is implemented as a web application and it has a 2-dimensional visualization tool to explore biological networks [1]. This tool has, however, some known scalability and visualization problems.

We have used MIxT in GeneNet VR as a case study. In addition, we used this case study in the evaluation since it is a realistic application where we used complex networks that originally didn't scale in the existing web application. In this chapter, we will describe what MIxT is used for, its disadvantages for scaling large biological networks, and the challenges of building a VR visualization system that can solve the visualization and scalability problems.

3.1 What is MIxT used for?

MIxT is a web application for interactive data exploration in system biology developed by UiT and Concordia University. The research was carried out for the study of interactions between the tumor and the blood systemic response of breast cancer patients. In the study, they profiled RNA in blood and matched

1. https://mixt-tumor-stroma.bci.mcgill.ca/network

tumor from 173 patients with breast cancer. The goal of the study was to identify genes and pathways in the primary tumor that are tightly linked to genes and pathways in the patient's systemic response (SR). The SR is the body's response to an infectious or non-infectious insult. A biological pathway is a series of actions among the molecules in a cell that leads to a certain product or change in the cell. The result of the study suggests new ways to monitor breast cancer by looking outside the tumor and studying the patient's systemic response.

MIxT provides an interactive view of networks. In Figure 3.1, we show a screenshot from the network view. This view is two-dimensional and the user can do some interactions to explore data, but these are very limited. The users can zoom in and zoom out and also drag and drop with the mouse in the view in order to "move around". Also, by hovering on a node, the user can see the name of that node. By clicking on a node, the user goes to another page with more detailed biological information about that gene.

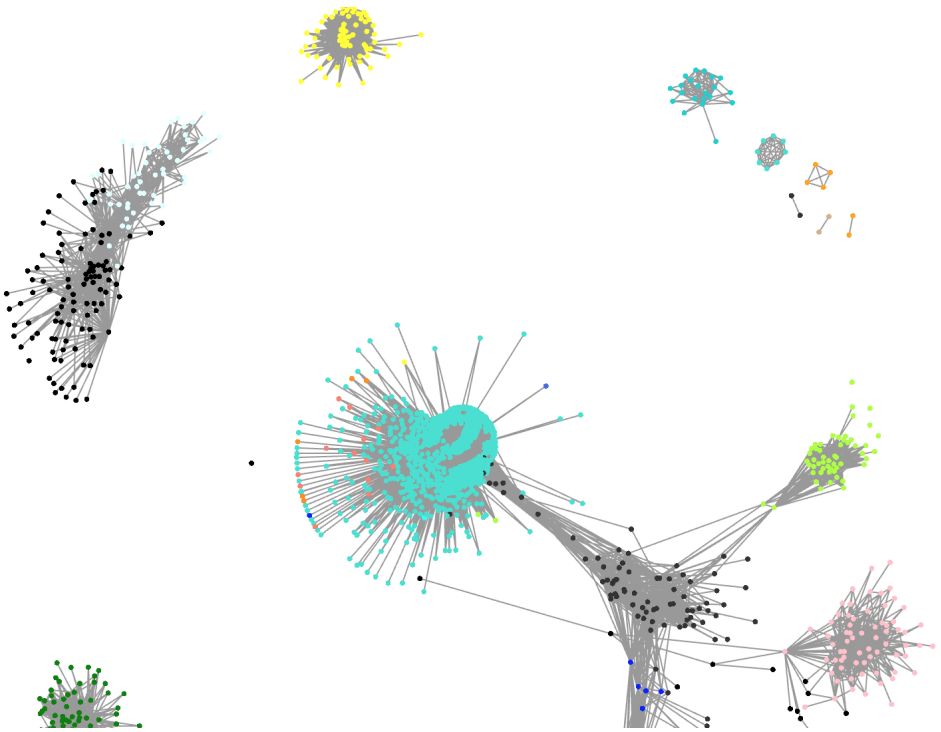


Figure 3.1: Screenshot from the network view in the MIxT web application.

An evident problem in this view appears when we start to explore the network. The clusters have many nodes and each node can have multiple edges, so the graph ends up looking like a hairball and it is not easy to explore (see Figure 3.2 for a screenshot of this problem from MIxT). In addition, the edges need

to be rendered every time the user interacts with the network. It may take a few seconds until the user can view all the edges that are in the view frame. This is a problem since the user needs to do many interactions when exploring the network. The visualization process becomes cumbersome and tedious in MIxT.

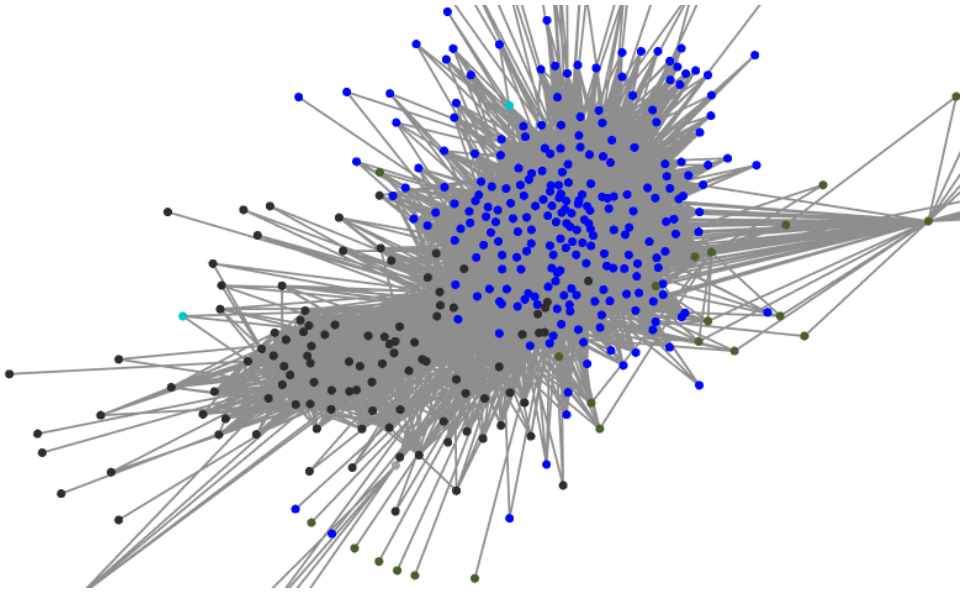


Figure 3.2: Hairball problem in the network view in MIxT.

3.2 MIxT in VR

We have implemented a Virtual Reality version of the network view from MIxT in order to solve some scalability and visualization problems. In addition to the original challenges that the network visualizer has, there are other challenges that we have to take into account in VR. In this section, we are going to explain these challenges and how we solved them in GeneNet VR.

When moving the original visualization system to VR, we have a new dimension in addition to immersion. Having a new dimension is an advantage when we want to visualize high-dimensional data, like biological networks. However, we need to cope with new problems like object occlusion. When the user visualizes the network from a particular angle, the nodes and edges that are in front of the user's viewpoint may hide other nodes and edges that are behind these. To solve this problem, we need to give the user the possibility to view the network from different angles. In GeneNet VR we have implemented a locomotion solution that allows the user to teleport to other parts of the scene. The user

can also rotate the viewpoint to the right or to the left. In addition, the user can also move the network around by using the VR controller.

As we mentioned before, the original network view from MIxT has some visualization problems. The network looks like a hairball and it's hard to explore it. In GeneNet VR we solve this problem by showing only the edges from the nodes that the user wants to explore. This solution reduces the amount of information that is shown, but it higher interactivity, so that the user can easily visualize the information. In GeneNet VR, we have implemented a node selector that allows the user to select a particular node using a laser pointer. When the laser collides with a node, the edges of this node are shown. Also, the user can scale up and down the network by using the VR controllers. In addition, a filtering menu was implemented in GeneNet VR, where the user can filter out the information from the network that is not relevant.

We have added the possibility to compare two datasets in real-time in GeneNet VR. This can be very useful for bioinformaticians when they work with several datasets like in MIxT. To compare two networks, the user can use a UI slider to visualize two datasets at the same time, creating what we call a morphing effect. To help the user distinguish from one dataset to another, we use linear interpolation for both the color of the nodes and their position.

3.3 Network characteristics

The networks from MIxT that we use in our case study are also known as gene co-expression networks (GCN) in biology. They are undirected graphs; meaning that the nodes can be connected together with bidirectional edges. In a GCN, the nodes represent genes and an edge between two nodes represents a significant co-expression relationship. As we previously showed in Figure 3.1, the nodes are in the networks are also organized in clusters and colors. Each cluster corresponds to a module, which is a subgraph where the genes are highly connected and where these genes are part of a common biological process that causes many interactions among themselves.

4

Evaluation and discussion

As part of the evaluation of our prototype, we wrote a list of questions focusing on performance and quality and that we will try to answer along with this chapter. These questions are:

1. For which interactions do we achieve the recommended FPS (72) for large biological networks?

2. What network properties influence the scalability?

3. Do we achieve the recommended FPS (72) for large biological networks when using the standalone Oculus Quest?

4. How do users perceive the visualization of large biological networks in GeneNet VR?

4.1 Methodology of experiment setup

GeneNet VR has been developed to explore large biological networks that contain genetic information. We have used two datasets from the MIxT project [7] and built a use case where we try to solve visualization and scalability problems.

The performance is an aspect of GeneNet VR that we want to evaluate. Without a good performance, visualization tools like this one can become tedious to use. Also, smooth interactions are needed, so that the user can easily explore the networks to find information and patterns in them. We will evaluate the performance for the interactions that are commonly used when exploring networks in GeneNet VR and that involve manipulating the position or size of the network or showing the edges. These are translate, scale, and select nodes in the network. Because of the time limitations, we couldn't evaluate other interactions. We will also evaluate if the number of nodes and edges influence the scalability of the application. We will also study if there are bottlenecks and what is causing them. In Table 4.1, we show the elements that compose the networks that we used and how they can influence the scalability.

Element	Description
Clusters	The algorithm used to create the clusters are run during the initialization of the system before the user can start exploring the networks. This can be time-consuming because it involves many operations to process the text files (they have several thousands of lines). However, this is only processed once.
Nodes	Represented as 2D squares in the space, they consist of 2 triangles. They are always showing in the scene and their position change while scaling, translating, and morphing the network.
Lines	They represent relationships between the nodes. Every time a node is selected, line objects are created in the scene. They are 2-dimensional and consist of 2 points. Depending on the node we might need to render several hundreds of these lines in the scene.

Table 4.1: Elements of the network that have influence in the scalability.

On the website of Oculus, we can find a reference with the Oculus' performance baselines that an application should meet [33] and that we will follow during the evaluation. These are the following:

- 72 FPS for Oculus Quest (required by Oculus).
- 50-100 draw calls per frame.
- 50,000-100,000 triangles or vertices per frame.

We will also evaluate the performance of the application being run on the Oculus Quest hardware, and compare it with the performance in the PC. The

hardware of the Oculus Quest is not as powerful as the machine's hardware that was used for the development. We would like to know if the performance on the Oculus headset is good enough for the visualization of datasets like the ones from MIxT.

As for the hardware specification, we ran the experiments in a machine with Windows 10. In Table 4.2, we can see the hardware specification for the machine. The GPU of the machine is also specified in Table 4.3. The hardware specification of the Oculus Quest is shown in Table 4.4.

Processor	Intel(R) Xeon(R) CPU E3-1275 v6 @ 2.80GHz 3.79 GHz
RAM	64.0 GB
System type	64-bit Operating System

Table 4.2: Machine specification.

Adapter type	NVIDIA GeForce GTX 1080 Ti
ROPS	88
Memory size	11 GB

Table 4.3: GPU specification.

Panel Type	Dual OLED 1600x1440
Supported Refresh Rate	72Hz
Tracking	Inside out, 6DOF
CPU	Qualcomm® Snapdragon 835
GPU	Qualcomm® Adreno™ 540 GPU
Memory	4GB total

Table 4.4: Oculus Quest specifications.

We built a benchmark in Unity in order to run the experiments several times. We used the same version of Unity as in the prototype, version 2018.4.10f1. The 3D rendering API that we have used is OpenGL. In total, we ran each experiment 4 times and we showed an average of the results using tables and graphs. The experiments are coded in the benchmark using scripts so that we can reproduce them several times. For the network translation and network scale interactions, we use mathematical functions to translate the network around the scene and to scale the network up and down. For the node selection, we have chosen a set of nodes.

We measured the frame time (time the frame takes to render), to analyze the performance. In Unity, the frame time is stored in a variable named deltaTime from the Time class. In order to find bottlenecks, we used a profiling tool from Unity. A profiler is used to get an overview of the performance of the application. This gave us information about per-frame CPU performance metrics. In addition,

Unity also provides some metric information that we can be displayed in the Unity editor. We got information about the number of vertices in the scene and the number of triangles.

To evaluate the quality of GeneNet VR, we conducted a series of interviews with several employees from UiT involved in computer science and biology research projects. These interviews were conducted in an informal way and have the purpose of obtaining feedback about aspects like the research contribution in bioinformatics, the performance of the application and the interactions, the usability, and also improvements that can be done to the project.

4.2 For which interactions do we achieve the recommended FPS (72) for large biological networks?

The experiments from this section were run on the PC and we used the blood dataset from MIxT. We chose this dataset because it is the largest one. We also ran each experiment for different sizes of the dataset: the whole dataset (2693 nodes), half size of the dataset (1346 nodes), and a third part (897 nodes). We didn't run the experiments using larger datasets and also on the Oculus Quest due to lack of time.

Each experiment lasts for 700 frames, starting from frame number 501 until frame 1200 in the application's timeline. We start from frame number 501 because some slowness occurs during the first frames, due to the initialization of the GeneNet VR. In order to evaluate the experiments, we stored the frame time for each frame for a total of 700 frames. The frame time is the time that a frame takes to render. We use this value to determine if the application meets the 72 FPS. Once we obtain all the frame times, we extract the following averages for each experiment: the average frame time of all the frames, the average of the 0.25% worst times, and the average of the 1% worst times.

The average of all the frames is the most important value to look at because it can give us an idea of whether the experiment meets the required FPS or not. In a perfect application where all the frames last the same amount of time and where the frame rate is 72, each frame would last around 13.9 milliseconds. However, this is not the case in real applications. If the general average is above 13.9 milliseconds, it means that the experiment didn't pass the 72 FPS. By looking at the 0.25% worst frame time and the 1% worst frame time, we will get the frame with the worst time and the 7 frames with the worst times respectively (our experiments last for 700 frames). If we see that these frames

have much worse times, compared with the general average, we will use a profiler to see if there are bottlenecks and try to give a solution to them.

4.2.1 Translating the network meets the 72 FPS

In this experiment, we evaluate the performance when translating the network. In the benchmark, the network is moved around in the scene using a sine function in the y-axis and a constant function in the z-axis. For every frame, we update the y and z position of the network object. In Table 4.5, we can see the results that we obtained.

Dataset size	0.25% average low	1% average low	Average
size	20.11	12.17	6.55
size/2	22	12.79	6.5
size/3	23.28	13.52	6.5

Table 4.5: Performance results in milliseconds when translating the network.

The averages for all the sizes are below 13.9 milliseconds, so translating the blood dataset network meets the 72 FPS. There isn't a significant difference between the averages for the different network sizes. This is because the network that we are using is probably too small. It would be interesting to use larger networks to find out which network size the performance is no longer good enough.

4.2.2 Scaling the network meets the 72 FPS

We also evaluated the performance when scaling the network. We used a sine function as well and update the size of the network object for every frame to scale it up and down. In Table 4.6 we can see the results that we obtained.

Dataset size	0.25% average low	1% average low	Average
size	22.82	13.42	6.51
size/2	22.61	12.69	6.49
size/3	23.02	13.63	6.49

Table 4.6: Performance results in milliseconds for the scale the network interaction.

The average time for this experiment is also under the 13.9 milliseconds limit, meaning that the interaction reaches 72 FPS. The low 1% is also under 13.9 milliseconds, so we don't consider that there are any big issues for the performance of this interaction. The variation among the network sizes is also insignificant. An evaluation with bigger networks would be necessary to determine the

network size limit for which the performance is not good.

4.2.3 Selecting nodes the network meets the 72 FPS but with possible bottlenecks

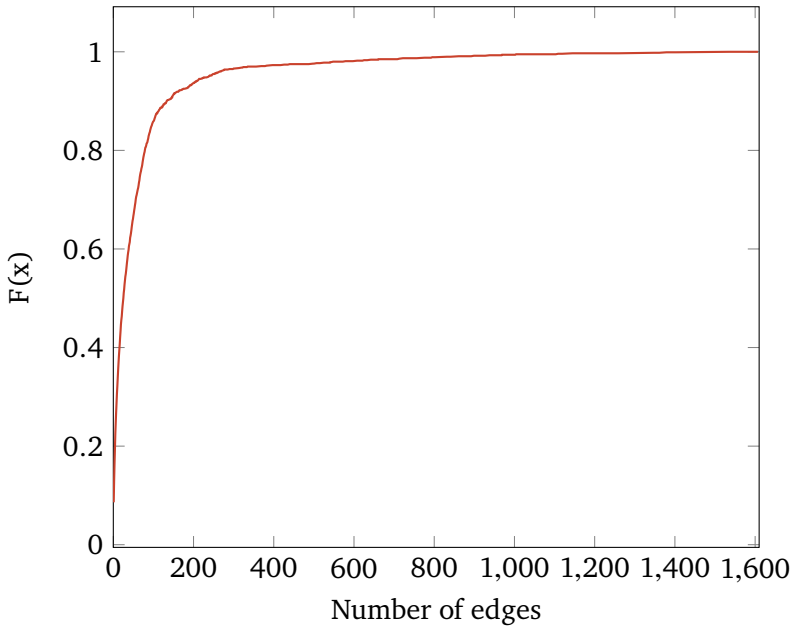


Figure 4.1: Distribution of the number of edges in the blood dataset. The x-axis shows the number of edges and the y-axes shows the distribution.

In this experiment, we want to evaluate the performance when selecting nodes in the network. When a node is selected in GeneNet VR, several things happen during this process. First, an algorithm finds the node that the user is trying to select. Second, two text objects are updated with the new names of the selected gene node. Third, the edges for the selected node are created in the scene.

In this experiment, we only evaluate the edge creation for the nodes, although it would be ideal to evaluate all the previous three parts. In the blood dataset, a node can have between 1 and 1607 edges. We wanted to have a balanced number of this for our experiment, so we decided to select several nodes that cover this range.

In Figure 4.1 we show the distribution of the number of edges in the blood dataset. As we can see, most of the nodes have less than 200 edges. For instance, there are 117 nodes that have 2 edges, and 231 nodes that have just 1 edge. Most of the nodes have very few connections. However, there a few nodes that have

several hundred of lines. It was a bit hard to make a balanced selection, so we just selected several nodes from the range that goes from 1 to 1607 (edges), even though this is probably unrealistic. The nodes that we select during the experiment are the following (in parenthesis the number of edges that they have): TGFBR3 (1), EPSTI1(11), SMNDC1(90), HNRNPH3(290), ANGEL2(586), ACTR6(756), ARGLU1(1607).

In Figure 4.7, we can see the results from this experiment.

Dataset size	0.25% average low	1% average low	Average
size	100	65.57	8.71
size/2	100	56.39	8.64
size/3	100	56.39	8.58

Table 4.7: Performance results in milliseconds for the select node interaction.

The average delta time is a bit higher than the ones in the translation and scaling experiments (around 2 milliseconds more), but still inside of the 72 FPS (under 13.9 milliseconds). For the low percentages, we don't reach the 72 FPS though. The results say that the creation of the edges when selecting a node can have more impact on the performance, and it would be interesting to find if there are any bottlenecks and how to solve them.

4.2.4 Performance discussion

From the results, we can conclude that there wasn't much point comparing the three sizes that we compared for the blood network, since the system is being underutilized. It would be interesting to evaluate larger networks where the system has performance problems in order to determine which size limits the system performs at 72 FPS.

The performance is good for the interactions that we evaluated. Having an average of 7-8 milliseconds means that we meet the 72 FPS that the Oculus' performance guidelines suggest. As we mentioned before, an average above 13.9 milliseconds is the limit. However, for the node selection, we have seen that the performance can be much worse than for the other interactions, see Figure 4.2 for a summary of the 1% worst results. The performance for this interaction is important because this shows relevant information to the user that is needed to explore the dataset. The edges are created in the scene every time a node is selected. A better solution could be to create the necessary edges in the scene when the application is initialized and show them where they are needed.

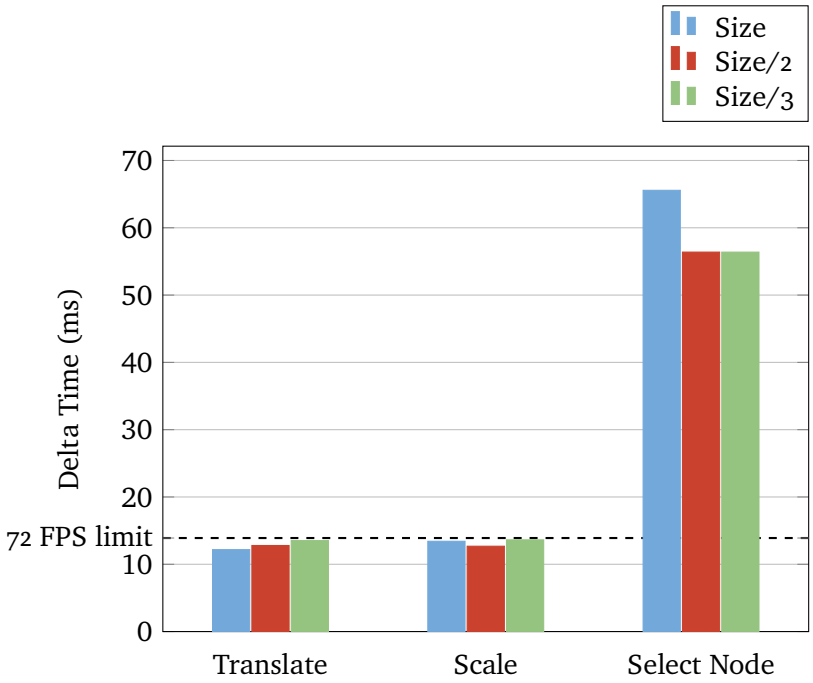


Figure 4.2: Bar graph showing a summary of the performance results for the 1% lowest average (7 frames with worst performance).

One important observation is that in the experiment we just select 7 nodes, one every 100 frames. However, when we use GeneNete VR, it's easy to select many nodes in just a few frames. This is because the select node function is run for every frame when we trigger the action for it, and if we point at a region where there are many nodes together, it will be easy to select several of them in a short period of time. Also, not all nodes have the same number of edges. It would be interesting to know if the number of edges could decrease the performance as well. We will analyze in the following section if the number of edges could have more impact on the performance and scalability of the network.

4.3 What network properties influence the scalability?

The number of nodes and the number of edges can influence the scalability of the system. We saw in the performance experiments that the datasets that we are using are small for our system. However, the time to create the edges when selecting the nodes can be very high. We will further evaluate the performance

when selecting the nodes in this section in order to determine how the number of edges can influence the scalability. For future experiments, it would be necessary to use networks with a higher number of nodes to understand how the number of nodes can influence scalability.

We designed an experiment where we test the time frame for different a amount of edges to render. In Figure 4.1 we showed the distribution between the number of edges and the number of nodes that have that amount of edges. We used this data and created a new dataset where we have a node from every "edge category" and then we selected each of the nodes and calculated the time frame. To put it simply, we try to render the edges for each of the possible numbers of edges that we can find in the blood dataset and calculate the time that GeneNet VR takes to create them in the scene.

In Figure 4.3, we can see the results as a scatter plot for this experiment. In the benchmark, we run the select node action for each of the nodes from the dataset that we created for this. We select a new node in every frame.

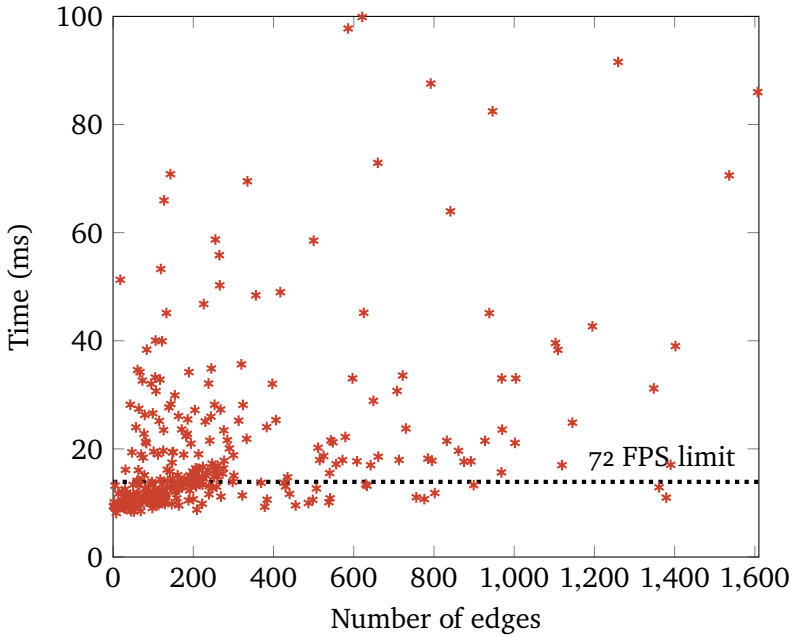
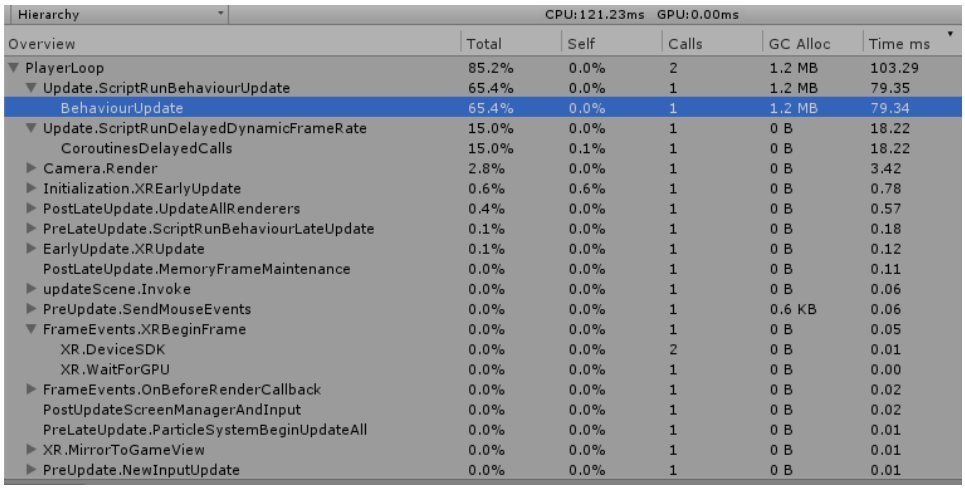


Figure 4.3: Scatter plot showing the relation between the number of edges to render and the time that it takes to render for the blood dataset.

From the results, we would have expected that when the number of edges to create is higher, more time is needed to create them. However, that is not what the results are showing. We can see that when there are less than 200 edges, many of the times are less than 20ms. Nevertheless, some nodes that have less than 200 edges take more than 40 or even 50 ms. On the right side of the plot,



Overview	Total	Self	Calls	GC Alloc	Time ms
PlayerLoop	85.2%	0.0%	2	1.2 MB	103.29
Update.ScriptRunBehaviourUpdate	65.4%	0.0%	1	1.2 MB	79.35
BehaviourUpdate	65.4%	0.0%	1	1.2 MB	79.34
Update.ScriptRunDelayedDynamicFrameRate	15.0%	0.0%	1	0 B	18.22
CoroutinesDelayedCalls	15.0%	0.1%	1	0 B	18.22
Camera.Render	2.8%	0.0%	1	0 B	3.42
Initialization.XREarlyUpdate	0.6%	0.6%	1	0 B	0.78
PostLateUpdate.UpdateAllRenderers	0.4%	0.0%	1	0 B	0.57
PreLateUpdate.ScriptRunBehaviourLateUpdate	0.1%	0.0%	1	0 B	0.18
EarlyUpdate.XRUpdate	0.1%	0.0%	1	0 B	0.12
PostLateUpdate.MemoryFrameMaintenance	0.0%	0.0%	1	0 B	0.11
updateScene.Invoke	0.0%	0.0%	1	0 B	0.06
PreUpdate.SendMouseEvents	0.0%	0.0%	1	0.6 KB	0.06
FrameEvents.XRBeginFrame	0.0%	0.0%	1	0 B	0.05
XR.DeviceSDK	0.0%	0.0%	2	0 B	0.01
XR.WaitForGPU	0.0%	0.0%	1	0 B	0.00
FrameEvents.OnBeforeRenderCallback	0.0%	0.0%	1	0 B	0.02
PostUpdateScreenManagerAndInput	0.0%	0.0%	1	0 B	0.02
PreLateUpdate.ParticleSystemBeginUpdateAll	0.0%	0.0%	1	0 B	0.01
XR.MirrorToGameView	0.0%	0.0%	1	0 B	0.01
PreUpdate.NewInputUpdate	0.0%	0.0%	1	0 B	0.01

Figure 4.4: Profiling the selection of the node ARGLU1(1607), which has the highest number of edges.

we can see that some nodes with several hundreds of edges take more than 60 ms to render, but in some cases, they take less than 20 ms. This is not very consistent and can conclude that the number of edges might influence in the performance and then in the scalability of the application, but we can see that there are many other aspects that influence the performance as well. Unity is a complex game engine, and there are many things going on when we run the experiments.

After running this experiment, we used the profiler from Unity to try to find the cause for taking so much time when selecting some nodes. We profiled the selection of the node ARGLU1(1607), which has the maximum number of edges. In Figure 4.4, we can see a screenshot of the profiler in Unity. There is a list of functions that are called during this frame. Also, there are several columns and we can see the percentage of the time that a particular function took. Analyzing the profiler, we can see that 65.4% of the total time comes from the BehaviourUpdate function. This function processes all the Update methods in Unity. It's probably that the problem is in the implementation that we chose. We would need further analysis into this and maybe split our function to create the lines into smaller functions and analyze each of them. But because of lack of time, we haven't done that here.

We get also some other metric information from the profiler such as the number of triangles that the scene is rendering. This is the selection of the node with the highest number of edges and there are a total of 35.2k vertices as we can see in Figure 4.5, which is inside of the Oculus' performance guidelines that we mentioned in this chapter.

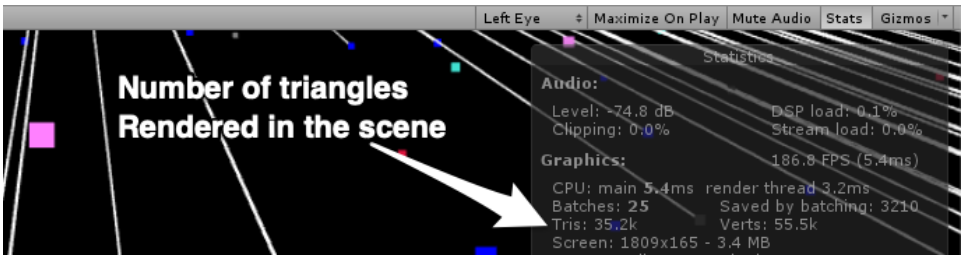


Figure 4.5: Number of triangles in the scene when selecting the node ARGLU1, which has 1607 edges, the largest number in the blood dataset.

It's hard to determine from the experiments that we ran if the number of edges can have an impact on the scalability of the system. We would need to better isolate this part of the code and further evaluate it. However, we expect that the number of edges can have an impact. As we have mentioned before, a better implementation solution could improve the performance of this. Instead of creating the edges in the scene every time a node is selected, we can create all the necessary edges in the initialization of the system, and show them when they are needed. So, normally they would be hidden, but we show them when the user selects a node.

4.4 Do we achieve the recommended FPS (72) for large biological networks when using the standalone Oculus Quest?

We would like to know if we can use a cheap VR headset like the Oculus Quest to visualize large biological networks. The hardware for the Oculus Quest is not as good as the machine's hardware, so we expect that the performance in the Oculus Quest will be worse. We also want to know how the performance is improved when using a machine with better hardware.

We designed an experiment that we could run in both the PC and the Oculus Quest to measure the performance. In this experiment, we used a combination of the three performance experiments for the interactions and we use them together for this experiment. For a short period of time, we translate the network, scale it, and select several nodes. The experiment lasts for 70 frames; it starts in frame 300 and ends in frame 370.

In Figure 4.6, we can see a graph with the results of the experiments. The performance for the Oculus Quest is worse than for the PC, as we had expected; however, most of the frames reach the 72 FPS limit. We can see some peaks

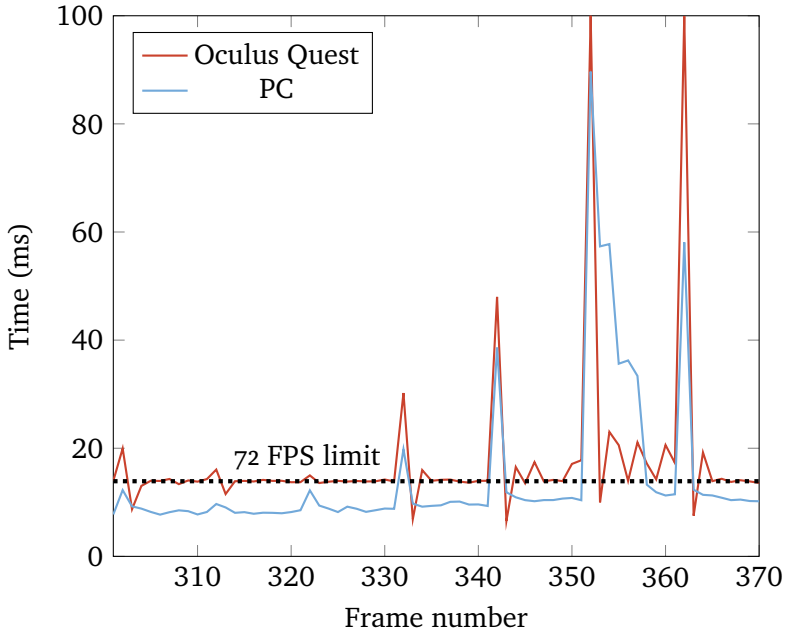


Figure 4.6: Performance of GeneNet VR when visualizing the blood dataset running on a machine and on the Oculus Quest. The x-axis represents the frame number (like a timeline). The y-axis represents the amount of time in milliseconds that a particular frame took to render.

in the graph, related to the selection of the nodes. We select a node every 10 frames (7 nodes in total). The nodes are selected in this order (in parenthesis is the number of edges): TGFBR3 (1), EPSTI1(11), SMNDC1(90), HNRNPH3(290), ANGEL2(586), and ACTR6(756).

From the results, we can say that we reach 72 FPS using the Oculus Quest for our network size. However, the performance is on the limit of 72 FPS and when we select nodes with more than 10 edges, the performance can get worse. It would be necessary to evaluate GeneNet VR using larger networks on the Oculus Quest. Also improve the creation of the edges as a better programming solution, as we mentioned before.

4.5 How do users perceive the visualization of large biological networks in GeneNet VR?

In order to answer this question, we used a qualitative research approach where we conducted a series of semi-structured interviews with several biologists and

computer scientists from the University of Tromsø. A semi-structured interview is an in-depth interview where the respondents have to answer open-ended questions [34]. The interviews were conducted only once and they covered the duration of 30 min to around one hour. We took handwritten notes during the interview to have the data captured. The open-ended questions that we used are the following:

- How do you perceive the application?
- How do you perceive the application for pattern finding?
- What is missing in the application?

Before starting the interview, a preliminary introduction to each respondent was given. We explained the research problem, as well as what MIxT is, the datasets that we are visualizing, and how they can use the different interactions with the VR controllers in GeneNet VR. Then the respondents tested the application using the Oculus Quest connected to the machine. We used the machine because there is a bug on the UI menu when running the application on the Oculus Quest hardware. In addition, with the Oculus Quest connected to the machine, we can see what the respondents are seeing in GeneNet VR and we can better guide them and have a more fluent communication. Due to the COVID-19 pandemic [35], we also applied some measures in the interviews. We tried to keep as much distance as possible and we also washed the VR headset and the controllers with sanitizer gel for each interview. We the open-ended questions to start conversations with the respondents where we discuss several quality aspects.

We interviewed 6 people in total. We gathered some information about their background as well. In Table 4.8 we have a summary of the participants and some information about their research field and if they have used VR before.

4.5.1 How do you perceive the application?

All the respondents think that GeneNet VR has some advantages for the visualization of large biological networks. The ones that hadn't used VR before or very little also think that Virtual Reality is an interesting way to visualize gene networks.

As for the performance of the application, none of the respondents found any performance issues in GeneNet VR. Also, they think that the interactions are smooth when exploring the networks. There was just a couple of times that

Respondent	Research field	Has used VR before?
1	Computer science and works with social networks	Owns an HTC Vive and uses on a regular basis
2	Clinical pharmacy and pharmacoepidemiology	Several times to play videogames
3	Pharmacoepidemiology. Has worked with drug networks	Has never used it
4	Biology. Has worked with gene co-expression networks	Has used it very little
5	Biology. Has worked with gene expression analysis	Has never user it
6	Biology. Has worked with GCN and biological networks	Has never used it

Table 4.8: Information about the respondents.

one of the interactions like the node selector didn't work, so we had to restart the application.

The participants who had used VR before easily learned how to use the different interactions with the controllers. The ones who hadn't used VR before or very little needed some more practice and guidance first but after a few interactions, they could feel comfortable exploring the networks. Respondent 1 also claimed that it would be necessary to use the application for some time in order to make the most of it. Respondents 4, 5, and 6 highlighted that the interactions are intuitive and easy to learn. However, respondent 1 claimed that it could be possible to implement more intuitive interactions. All the participants also think that the quantity of interactions seems good and that there are not too many to learn.

During the interviews, we also asked if using a standalone headset like the Oculus Quest is advantageous. Respondents 1 and 5 respondents said that it is. The reason why it is advantageous is that it is easier to use since you don't need cables and you can also stand and use it anywhere you want. It is also easy if you can walk around in order to explore the network, according to respondent 5. On the other hand, two respondents (2 and 3) claimed that they prefer to use this kind of visualization tool while sitting. The rest of the respondents didn't consider it relevant.

4.5.2 How do you perceive the application for pattern finding?

The participants agree that finding patterns in GeneNet VR seems to be easier than in other visualization tools that they have used. They also agree that they don't know the MIxT datasets very well so it can be hard to know what to look for. The participants with a biological background seemed to understand better the datasets and they knew some of the genes from the network. Some participants (1, 2, 3, and 4) said that it would be helpful to find patterns in their datasets. Participant 4 was also interested in visualizing a GCN dataset that the participant was working with at the moment. Participant 1 was also interested in visualizing social networks. The participants from the field in pharmacoepidemiology also claimed that it would be very helpful for them to visualize their drug datasets using GeneNet VR because the networks are very similar.

We also received some feedback about how to improve the visualization for pattern finding. For example, it would be useful to show the names of the connected nodes to the node that we are selecting. Also, the possibility to isolate or highlight the nodes that we are selecting would be helpful. Finally, pattern finding would also be easier if they had the chance to change the layout of the network.

4.5.3 What is missing in the application?

We obtained interesting feedback about how to improve GeneNet VR. Since we interviewed several research scientists from different fields, we also obtained feedback for the visualization of other networks like drug and social networks. These networks share a similar structure with the gene co-expression networks.

The edges in MIxT represent significant co-expression relationships between two nodes. This information is shown in GeneNet VR as a line that goes from one node to another. However, we don't show how significant co-expressed the nodes are. This is some relevant information that especially the biologists are missing. Most of the participants suggested using a color range that goes from less significant co-expressed to more significant co-expressed or by changing the thickness of the edges.

The names of the nodes that are connected to the selected node are also relevant information that is missing. As suggested from the interviews, we can show a small UI window with the list of those nodes, or a text label on them. A solution could also be to show the name of the connected node when we

get close to it or to show all of them but highlighting the nodes that we are selecting.

Some respondents (1, 3, and 4) mentioned that it would also be useful to rotate the networks using the hand controllers. The Oculus Quest headset is equipped with 2 6DOF controllers that support both orientation and positional tracking. Respondent 1 pointed out that this problem could be solved by using the orientation of the controllers.

We have created a list with other improvements for GeneNet VR that we discussed in the interviews:

- Improve the distance feeling. The nodes are 2-dimensional shapes, and they all have the same brightness. A solution could be to make them darker when they are far away.
- Possibility to change the layout of the network in order to categorize the arrangement of the nodes.
- Add favorite angles: from above, from below, from a particular part of the network.
- Filter the nodes by clusters.
- Instead of use snap rotation, use a smooth rotation.
- Change the shape of the nodes into spheres or into a more 3-dimensional shape.
- Search functionality to search for node names.
- Possibility to "pin" a node in order to compare it with other nodes.
- Use centrality measures: change the color by centrality.
- Possibility to turn on and off the name of the node and also the edges.
- Add a collision circle to the nodes so that they are easier to select.
- Have the possibility to use other pathway libraries.
- Possibility to manually add other genes.
- Use a slider to filter the nodes by their co-expression value.

4.5.4 Discussion

The results from the quality evaluation that we carried out indicate that GeneNet VR is a helpful visualization tool for large biological networks. It is easy to use and to learn even for users new to VR, the performance and the interactions are smooth and thanks to the interactions that GeneNet VR provides, the users can easily explore these networks.

GeneNet VR is also implemented for the Oculus Quest headset, which is a standalone headset that doesn't need cables. This has shown to be an advantage for some users who prefer more flexibility when they need to use a visualization tool. As respondent 1 mentioned, it is also encouraging because you can use it anywhere you want.

We would like to highlight from the results that all the participants except number 5 were very interested in visualizing their datasets in GeneNet VR because they would find it very helpful. Participant 4 works with some biological GCN datasets and has a hard time trying to extract patterns and conclusions from them. But GeneNet VR would solve many problems for participant 4 because it offers a better way to explore them and the process would be much faster. Participant 6 also works with some biological networks and would find it interesting to visualize the datasets with GeneNet VR. Participant 1 was interested in visualizing social networks because this tool will give the participant a better perspective in order to find interesting patterns. Also, the participants from pharmacoepidemiology work with drug networks and this tool would give them a better way to visualize these datasets.

5

Related work

I will focus in this chapter on similar works found in the literature for the visualization of bioinformatics data in VR.

5.1 Virtual Reality Chemical Space

Virtual Reality Chemical Space is a VR application for the interactive exploration of chemical space populated by Drugbank compounds[36]. It is also developed in Unity using C# and the VRTK library. They use a particle system to render the particles of the chemical space. To render the particles, they use shaders instead of geometrical spheres as well, optimizing the number of vertices per datapoint in the scene. In order to reduce motion sickness, they have introduced a floor in the form of a grid acting as a static frame of reference. Also instead of letting the user move through the VR environment, they use a controller to move the point cloud. In Figure 5.1 we can see a screenshot from the VR application. As we can notice in the screenshot, there is a subtle rendering of an outer space scene as an independent visual background. This is another technique to reduce motion sickness that helps the user keep the notion of space. This is an interesting technique that we could have tried in GeneNet VR.

They conclude in the article that the application that they have developed doesn't visualize an environment with an analog in the real world, but instead

a mathematical construct. Also, some of the drawbacks that VR has compared to traditional solutions is that the VR headsets are not so comfortable as well as often-occurring eye strain and virtual reality sickness. They also conclude that the application of VR in chemistry has more potential in the fields of education and training than for the current state of technology. The tools for chemistry need to be further evaluated whether they should be extended to VR.

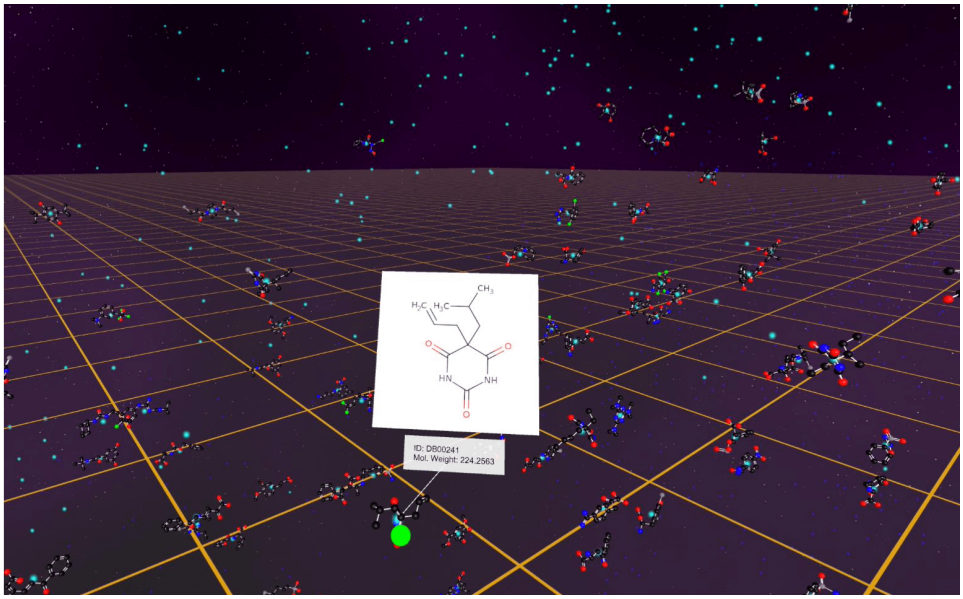


Figure 5.1: Optimized virtual reality chemical space. Figure taken from [36].

5.2 BioVR

BioVR is an interactive VR platform for integrated visual analysis of DNA/RNA protein structures[37]. It is built in Unity and using C#. The headset that they targeted the application to is Oculus Rift. One big difference between BioVR and our application is that in BioVR they use the hands for the interactions rather than the controllers. This can be very attractive and it could have worked very well for GeneNet VR since most of the interactions can be done with hand gestures. Also since early 2020, hand tracking has been integrated into Oculus Quest, making it easier for dveleopment[1]. A screenshot of BioVR can be seen in Figure 5.2. The user in BioVR can visualize the virtual hands in the virtual world and they are used for the interaction with the nucleotide sequence and UI menus. In GeneNet VR instead of the hands, we show the controllers, which help the user orientate in the space. The research concludes that using VR

1. https://developer.oculus.com/documentation/unity/unity-handtracking

helps in create new workflows for researchers to view DNA/RNA sequence and protein structures. Also, in VR it is easy to integrate 2D user interfaces in the virtual world, which can be very helful for users as we can see in Figure 5.2 B.

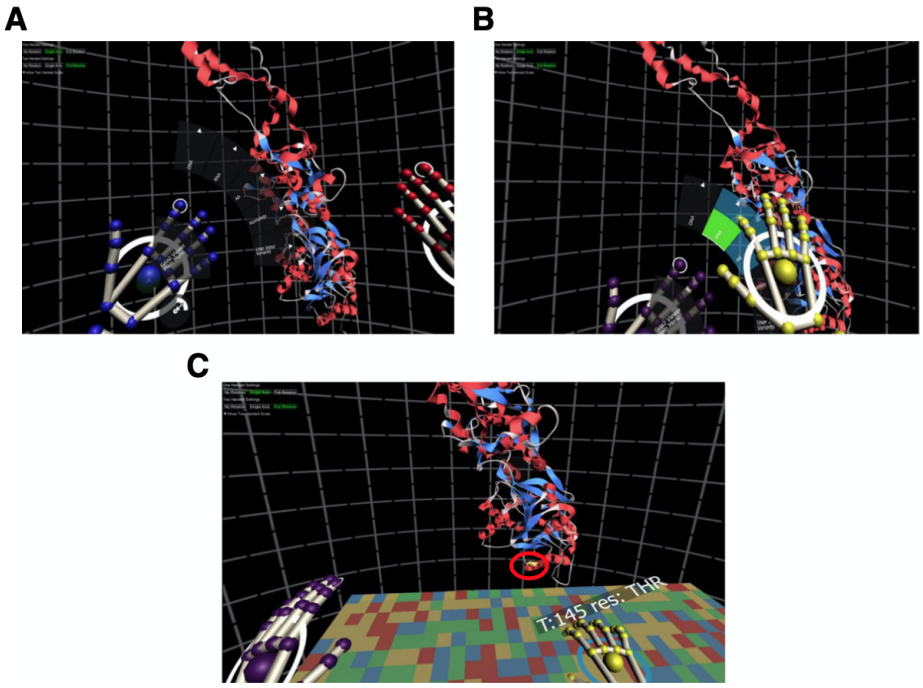


Figure 5.2: Screenshot from BioVR. Figure taken from [37].

5.3 CellexalVR

CellexalVR is a virtual reality environment for the visualization and analysis of single-cell RNAseq experiments that help researchers understand their data[38]. The system is divided into two parts: the first one consists of the VR interface and the second is an R package called cellexalvrR that does back-end calculations and also provides functions that allows the user to export the scRNAseq data from an R session for CellexalVR to read. CellexalVR was developed in Unity for HTC Vive (Pro). They used Unity and C# and R for the implementation. They used libraries like VRTK, OpenVR, and SteamVR as well in the implementation.

Something that is interesting in CellexalVR is that it allows a multi-user mode via the Photon Unity Networking. This works by sending information about the events to each user using Remote Procedure Calls. In Figure 5.3 we can see

a screenshot from CellexalVR where two users participate in the same session. Other users can also be in the same session but just be watchers and be in like a "ghost mode". This is something interesting that could be used in GeneNet VR as well. Especially the "ghost mode" could be useful so that other people can visualize the network at the same time from other perspectives.

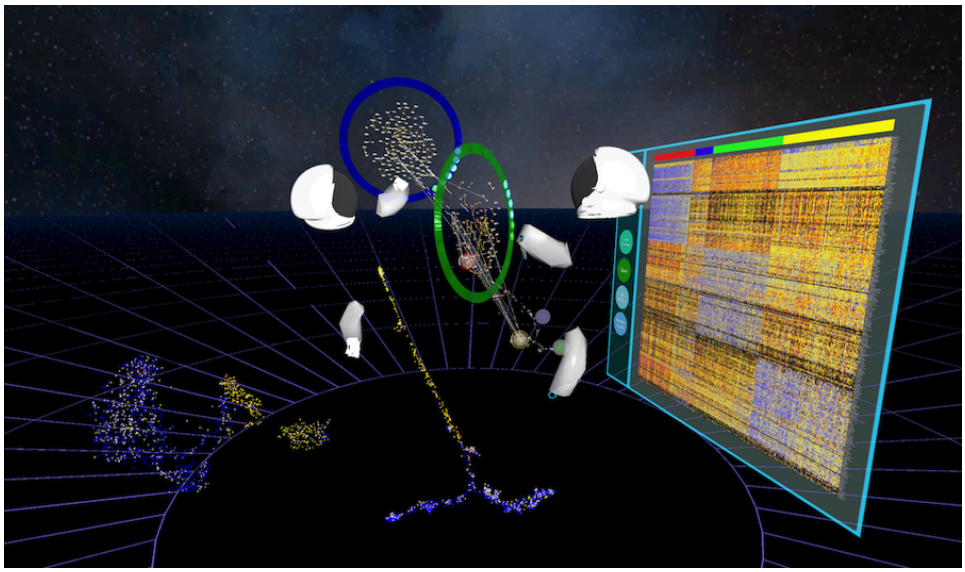


Figure 5.3: Screenshot from CellexalVR. Two users using CellexalVR at the same time. The head models were taken from NASA. Figure taken from [38].

5.4 BigTop

BigTop is a visualization framework in VR for the rendering of Manhattan plots in three dimensions[39]. Manhattan plots are usually 2-dimensional, where genomic coordinates are displayed on the x-axis and the negative log-10 of the association P-value for each single nucleotide polymorphism (SNP) displayed on the Y-axis. Each dot on the Manhattan plot signifies an SNP then. In BigTop, the z-axis is used to display the minor allele frequency of each SNP. This allows the identification of allelic variants of genes. As for the interaction, BigTop allows the user to select a node in order to obtain more information like the SNP name. BigTop is built in JavaScript with the React and A-Frame frameworks. It can also be rendered in any commercially available VR headsets and also in 2D web browsers.

To move around the scene in BigTop the user can take steps (in the VR version) or using the arrow keys from the keyboard. The user can select the nodes by pointing with a laser at the nodes in the scene in the VR version (See Figure

5.4 for a screenshot of BigTop showing the selection of a node). In the browser version, it is possible to select them using the pointer from the mouse. Also in order to look around, the user needs to move the head around using the VR headset, or by using click and drag with the mouse (in the browser). In GeneNet VR we have focused only on building a visualization system for a VR headset. We have used better locomotion techniques that improve the comfortability. However, the locomotion technique that they use to move around could be a good idea for GeneNet VR as well.

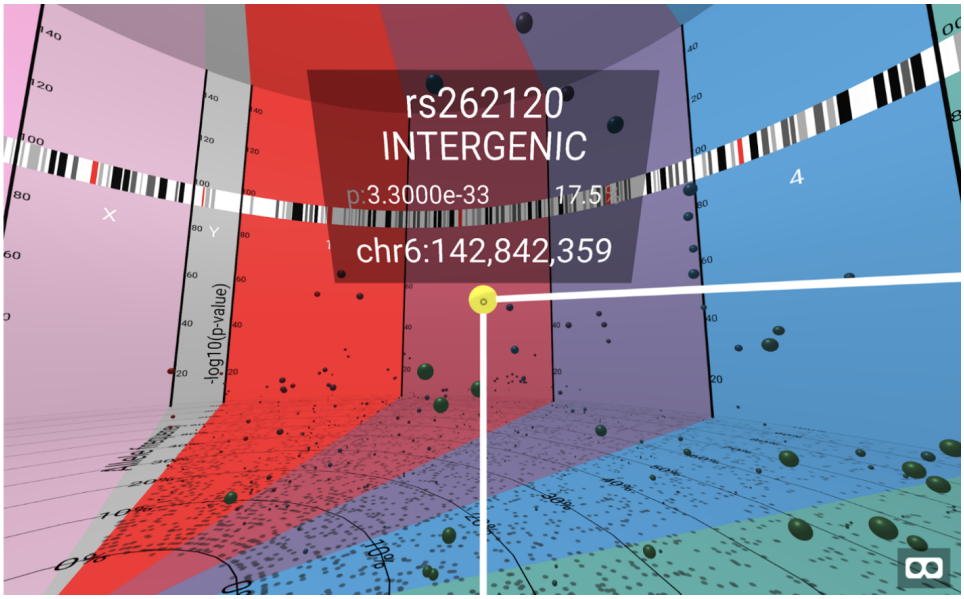


Figure 5.4: Screenshot from BigTop where a node is selected. Figure taken from [39].

5.5 Unity vs Unreal Engine

Unity3D[2] and Unreal Engine[3] are two popular software for the development of videogames as well as virtual reality games and other applications. They offer integration for Oculus Quest and other VR devices in the market.

I chose Unity for the development mostly because I had some experience with it. In addition, when I researched VR development, I found that Unity is well integrated with Oculus Quest and there is the VRTK library. I also found several tutorials about VR development in Unity as well as documentation for the Oculus integration and about VRTK. Unity is also known to be easier to learn. There is a study where they teach Unity and Unreal Engine to students,

2. https://unity.com
3. https://www.unrealengine.com

and they concluded that Unity is a little less complicated to learn [40]. On the other hand, Unreal can create more professional-looking results. For this project, we are not looking for realistic graphics, since we are dealing with abstract data.

5.6 VR toolkits and frameworks

Apart from the Oculus integration for Unity, there are some 3rd party toolkits and frameworks that we can use to accelerate the development process for VR. I used VRTK in my project, as I have mentioned before. This toolkit contains several scripts and prefabs that can be used to build several VR solutions. In my case, I used this library to build the teleportation and snap rotation locomotion techniques. VR Interaction Framework[4] is another toolkit for VR development, but it is not open source. In addition, Unity comes with XR, support for XR platform integration.

5.7 VR headsets

We can find numerous VR headsets nowadays in the market[5], from simple headset solutions like Google Cardboard to more advanced ones like HTC Vive Focus. In order to choose which headset we wanted to target our application to, we had into account the price, performance, and features that the headsets have. We chose Oculus Quest because the price was not very high (5799,- kr in 2019) and it was also a standalone device, which is not so common among the VR headsets. Oculus Quest also has an advantage as for the performance, it can be connected to a PC where we can run heavier applications. In this way, this headset is very versatile and the price is affordable for many people and especially for research entities. Other features that Oculus Quest has is that it comes with two controllers that have free movement and inputs as buttons, triggers and grips that can be used for the interactions in the VR application. Other headsets from a lower price range like the Oculus Go have fewer inputs in the controllers and are more stationary.

4. http://beardedninjagames.com/vr-framework
5. https://versus.com/en/vr-headset

6

Conclusion

We have developed GeneNet VR, a Virtual Reality application for the visualization of large biological networks. We used two datasets from MIxT, a real application with visualization problems, as a use case and solved scalability and information overload problems. GeneNet VR gives bioinformaticians the possibility to explore large biological networks for pattern identification using the Oculus Quest, a standalone economical VR headset.

Previous work for the visualization of large biological networks has shown common problems like the hairball problem, cumbersome interactions and scalability issues. We solve these in GeneNet VR by taking advantage of the rich interactivity that VR offers in order to balance out the amount of information. We have implemented several interaction and visualization techniques that help the users explore the large datasets. A locomotion system is used, providing the user a way to move around in the scene, also solving object occlusion problems common in 3D spaces. The user can also move the network around, zoom in and out, and filter the nodes using a -dimensional UI. The edges of the network are shown for each node when the user selects them using a laser pointer. Also, a novel feature allows the user to compare two datasets in simultaneously.

We ran several performance experiments for our case study and demonstrated that GeneNet VR performs well when exploring the networks from MIxT. We also evaluated several interactions that are commonly used during the visualization process and achieved the required FPS required by Oculus. Likewise,

we evaluated the performance on the Oculus Quest hardware and the results indicated that the Oculus Quest is around 30% slower than on the PC, but it still reaches the 72 FPS.

We evaluated the quality of of our project with a qualitative research approach where we conducted purposive sampling with in-depth semi-structured interviews with research scientists from UiT. We obtained very good outstanding results, where the respondents highlighted that GeneNet VR is an interesting and useful tool to explore biological networks. The system is also easy to learn for the respondents, even for those new to VR, and the interactivity is smooth, making it easy to explore the networks. We also made a list with further requirements that we extracted from the interviews that will help us improve GeneNet VR in the future.

We believe that GeneNet VR is an important and useful tool for bioinformaticians. We have traced out the guidelines that facilitate VR as an affordable and advantageous option for the visualization of large biological networks. The rich interactivity that VR offers and the advancement in hardware has allowed us to solve visualization and scalability problems that are common in similar visualization tools. Now we ask ourselves, what are the next steps of GeneNet VR? Furthermore, can we use a similar solution to visualize similar networks from other fields? Thanks to the interviews that we carried out, we believe that GeneNet VR has potential in our research problem and can certainly be useful in other fields, such as for the visualization of drug and social networks.

7

Future work

GeneNet VR has room for improvements and new ideas. During the development, I used a git repository hosted on GitHub. I used the issues list that GitHub offers in order to have an overview of the problems that the application had. There are still a few issues opened and that I didn't have time to work with; some are bugs and others are ideas for future development. The project is now open source and anyone can contribute to it. The repository can be found in this URL on my GitHub account [1]. The issue list can be found in this URL [2].

As we can see in the list, there are some issues that are tagged as "bug"; These bugs are not very big and don't prevent GeneNet VR from being functional. However, they have some impact on the usability of the application.

A problem that we found out is that when the application is run on the Oculus Quest hardware, the pointer for the 2D menu doesn't work. With this problem, the user cannot use the filters or the morph slider. Also when we morph the datasets, the new position of the networks is not taken into account if they have been translated previously. Other small bugs are that there seems to be an offset in the teleportation arrow, so the user doesn't teleport exactly to the direction of the arrow; also sometimes the pointer selects nodes that are behind the users, and it should be only the ones that are in front of the user.

1. https://github.com/kolibrid/GeneNet-VR
2. https://github.com/kolibrid/GeneNet-VR/issues

We believe that GeneNet VR has also the potential for new development. We have written som ideas and tagged them with the word "enhancement" on GitHub. Something that could be interesting is to make the application multiuser. We got this idea from CellexalVR[38], where two people can be in the same session and only one of them can interact with the data, the other is only a watcher. We could implement something similar in GeneNet VR, where one of the users is just a watcher but can for instance teleport to other parts of the network to visualize it.

Another improvement that we think that would enhance the performance of the application, is to create the lines that correspond to the edges during the initialization of the application and show them when they are needed. Right now, these lines are created in the scene every time a new node is selected. We use a function from Unity called Instantiate for this, which can downgrade the performance of the application, according to some forums from Unity.

Other improvements for BigNet VR would be focused on the aesthetics of the application. It's something that we haven't focused much in our prototype. An issue that we can see is that the nodes that we filter and that are morphed are turned into a black colour. It would be better to make the nodes transparent or to remove them from the network (when we filter them). If the nodes are still there but in black colour, they could hide other nodes that are behind them and the user might not see those nodes.

Other ideas that we had to improve GeneNet VR are for example highlight the nodes that are selected and their related nodes, the possibility to adjust the space between the nodes in the clusters and improve the 2D elements like the menu.

Finally, we would also like to run other experiments to further evaluate the performance of the application and study better the scalability. We would like to use larger datasets in order to determine where is the size limit for which we can visualize these datasets in GeneNet VR for both PC and the Oculus Quest headset.

7.1 New requirements based on the interviews

We obtained very interesting feedback from the interviews. We have created a list of issues based on this feedback and tagged them with the word "interviews" on GitHub.

Some respondents from the interviews who work on biology research projects

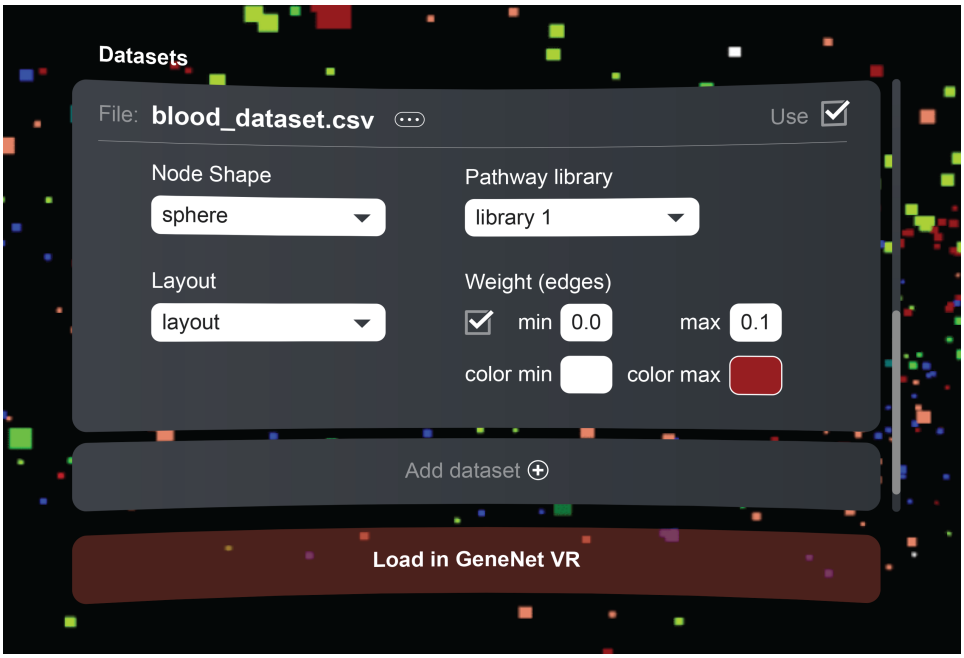


Figure 7.1: Design of a user interface for GeneNet VR with help of a graphic designer.

claimed that it would be helpful to also see the names of the nodes that are interconnected to the selected node. When we select a node, we only get information for the node name that we have selected, but not for the interconnected nodes. A solution to this is to show a text label on the interconnected nodes or a small window with the interconnected node names. This task is tricky because we can end up having a lot of information. We can also try to highlight the nodes that we are visualizing and show the name labels when we get close to the nodes.

There is also information in the edges that we are not showing, as the weights. This information is common in networks like co-expression gene, social and drug networks. We can show the weight values by using a colour range in the lines or by changing the thickness of these. The distance between one and another node can also give important information about this.

Other improvements that we would like to do based on the interviews is: to be able to rotate the network with the controllers, filter the nodes by clusters, use a node search functionality, make the snap rotation smoother, be able to pin a node to compare it with other nodes and make it easier to select nodes by showing a collision circle around the nodes.

We also discussed the idea of building a notebook interface where the bioin-

formaticians can edit the datasets before uploading them to GeneNet VR. A notebook interface is a virtual notebook environment used for literate programming. It's a word processing tool where we also have a shell and kernel of a programming language. Notebooks are very useful to analyze data. The idea would be that the users can edit the dataset by changing the layout, using a different pathway library, changing the co-expression limit, adding other nodes, etc. then GeneNet CR is used to visualize the dataset by using the interaction functionalities.

Finally, we have created an improved user interface with some UI elements for GeneNet VR with help of a graphic designer, as we can see in Figure 7.1. These UI elements could be used in other parts of GeneNet VR like the filtering menu, a node search menu or to edit properties of the current network, to cite some examples.

Bibliography

[1] Mike May. "LIFE SCIENCE TECHNOLOGIES: Big biological impacts from big data." In: *Science* 344.6189 (2014), pp. 1298–1300. ISSN: 0036-8075. DOI: 10.1126/science.344.6189.1298. eprint: https://science.sciencemag.org/content. URL: https://science.sciencemag.org/content/344/6189/1298.

[2] Jimmy F. Zhang et al. "BioVR: a platform for virtual reality assisted biological data integration and visualization." In: *BMC Bioinformatics* 20.1 (2019). DOI: 10.1186/s12859-019-2666-z.

[3] Keiichiro Ono. URL: https://cytoscape.org/.

[4] Kevin R. Brown et al. "NAViGaTOR: Network Analysis, Visualization and Graphing Toronto." In: *Bioinformatics* 25.24 (2009), 3327–3329. DOI: 10.1093/bioinformatics/btp595.

[5] *BioLayout Express 3D for visualising biological networks*. URL: https://www.ebi.ac.uk/about/news/service-news/BioLayoutExpress3D.

[6] Giuseppe Agapito, Pietro Guzzi, and Mario Cannataro. "Visualization of protein interaction networks: problems and solutions." In: *BMC Bioinformatics* 14.Suppl 1 (2013). DOI: 10.1186/1471-2105-14-s1-s1.

[7] Vanessa Dumeaux et al. "Interactions between the tumor and the blood systemic response of breast cancer patients." In: *PLOS Computational Biology* 13.9 (2017). DOI: 10.1371/journal.pcbi.1005680.

[8] Miguel Angel Pujana et al. "Network modeling links breast cancer susceptibility and centrosome dysfunction." In: *Nature Genetics* 39.11 (2007), 1338–1349. DOI: 10.1038/ng.2007.2.

[9] Andrew G Fraser and Edward M Marcotte. "A probabilistic view of gene function." In: *Nature Genetics* 36.6 (2004), 559–564. DOI: 10.1038/ng1370.

[10] Daniele Merico, David Gfeller, and Gary D Bader. "How to visually interpret biological data using networks." In: *Nature Biotechnology* 27.10 (2009), 921–924. DOI: 10.1038/nbt.1567.

[11] Bjørn Fjukstad et al. "Building Applications for Interactive Data Exploration in Systems Biology." In: *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics - ACM-BCB 17* (2017). DOI: 10.1145/3107411.3107481.

[12] Guangyan Zhou and Jianguo Xia. "OmicsNet: a web-based tool for creation and visual analysis of biological networks in 3D space." In: *Nucleic Acids Research* 46.W1 (2018). DOI: 10.1093/nar/gky510.

[13] H. Kilicoglu TC. Rindflesch et al. "Network-based analysis reveals distinct association patterns in a semantic MEDLINE-based drug-disease-gene network." In: (1970). URL: https://link.springer.com/article/10.1186/2041-1480-5-33.

[14] KE. Laver et al. "Virtual reality for stroke rehabilitation." In: *Cochrane Database of Systematic Reviews* 11 (2017). ISSN: 1465-1858. DOI: 10.1002/14651858.CD008349.pub4. URL: https://doi.org//10.1002/14651858.CD008349.pub4.

[15] James Xia et al. "Three-dimensional virtual-reality surgical planning and soft-tissue prediction for orthognathic surgery." In: *IEEE Transactions on Information Technology in Biomedicine* 5.2 (2001), 97–107. DOI: 10.1109/4233.924800.

[16] "Commentary on Rose, F.D., Brooks, B.M., & Rizzo, A.A., Virtual Reality in Brain Damage Rehabilitation: Review." In: *CyberPsychology & Behavior* 8.3 (2005), 263–271. DOI: 10.1089/cpb.2005.8.263.

[17] Yuting Yang et al. "Integration of metabolic networks and gene expression in virtual reality." In: *Bioinformatics* 21.18 (July 2005), pp. 3645–3650. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bti581. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/21/18/3645/520673/bti581.pdf. URL: https://doi.org/10.1093/bioinformatics/bti581.

[18] David A. Thorley-Lawson, Karen A. Duca, and Michael Shapiro. "Epstein-Barr virus: a paradigm for persistent infection – for real and in virtual reality." In: *Trends in Immunology* 29.4 (2008), 195–201. DOI: 10.1016/j.it.2008.01.006.

[19] Corey J. Bohil, Bradly Alicea, and Frank A. Biocca. "Virtual reality in neuroscience research and therapy." In: *Nature Reviews Neuroscience* 12.12 (2011), 752–762. DOI: 10.1038/nrn3122.

[20] Matthias Minderer et al. "Virtual reality explored." In: *Nature* 533.7603 (2016), 324–325. DOI: 10.1038/nature17899.

[21] *OVRInput*. URL: https://developer.oculus.com/documentation/unity/unity-ovrinput/.

[22] Evren Bozgeyikli et al. "Point & Teleport Locomotion Technique for Virtual Reality." In: *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY 16* (2016). DOI: 10.1145/2967934.2968105.

[23] R.a. Ruddle. "The effect of environment characteristics and user interaction on levels of virtual environment sickness." In: *IEEE Virtual Reality 2004* (). DOI: 10.1109/vr.2004.1310067.

[24] *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. URL: https://tools.ietf.org/html/rfc4180.

[25] Unity Technologies. *ParticleSystem*. URL: https://docs.unity3d.com/ScriptReference/ParticleSystem.html.

[26] Unity Technologies. *Line Renderer*. URL: https://docs.unity3d.com/Manual/class-LineRenderer.html.

[27] Unity Technologies. *Prefabs*. URL: https://docs.unity3d.com/Manual/Prefabs.html.

[28] *Unity 2018.4.10*. URL: https://unity3d.com/unity/whats-new/2018.4.10.

[29] Jason Jerald et al. "Developing virtual reality applications with Unity." In: *2014 IEEE Virtual Reality (VR)* (2014). DOI: 10.1109/vr.2014.6802117.

[30] *Unity Integration*. URL: https://developer.oculus.com/downloads/package/unity-integration/1.29.0/.

[31] *Welcome to VRTK · VRTK - Virtual Reality Toolkit*. URL: https://vrtoolkit.readme.io/docs/summary.

[32] Oculus. *Oculus Link Compatibility*. URL: https://support.oculus.com/444256562873335/.

[33] *Testing and Performance Analysis*. URL: https://developer.oculus.com/documentation/unity/unity-perf/?device=QUEST.

[34] Shazia Jamshed. "Qualitative research method-interviewing and observation." In: *Journal of Basic and Clinical Pharmacy* 5.4 (2014), p. 87. DOI: 10.4103/0976-0105.141942.

[35] *Coronavirus*. URL: https://www.who.int/emergencies/diseases/novel-coronavirus-2019.

[36] Daniel Probst and Jean-Louis Reymond. "Exploring Drugbank in Virtual Reality Chemical Space." In: (2018). DOI: 10.26434/chemrxiv.6629150.

[37] Jimmy F Zhang et al. "BioVR: a platform for virtual reality assisted biological data integration and visualization." In: (2018). DOI: 10.1101/307769.

[38] Oscar Legetth et al. "CellexalVR: A virtual reality platform for the visualisation and analysis of single-cell gene expression data." In: *bioRxiv* (2019). DOI: 10.1101/329102. eprint: https://www.biorxiv.org/content/early/2019/07/16/329102.full.pdf. URL: https://www.biorxiv.org/content/early/2019/07/16/329102.

[39] Samuel T. Westreich, Maria Nattestad, and Christopher Meyer. "BigTop: A Three-Dimensional Virtual Reality tool for GWAS Visualization." In: (2019). DOI: 10.1101/650176.

[40] Paul E. Dickson et al. "An Experience-based Comparison of Unity and Unreal for a Stand-alone 3D Game Development Course." In: *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education* (2017). DOI: 10.1145/3059009.3059013.