UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Faculty of Science and Technology
Department of Computer Science

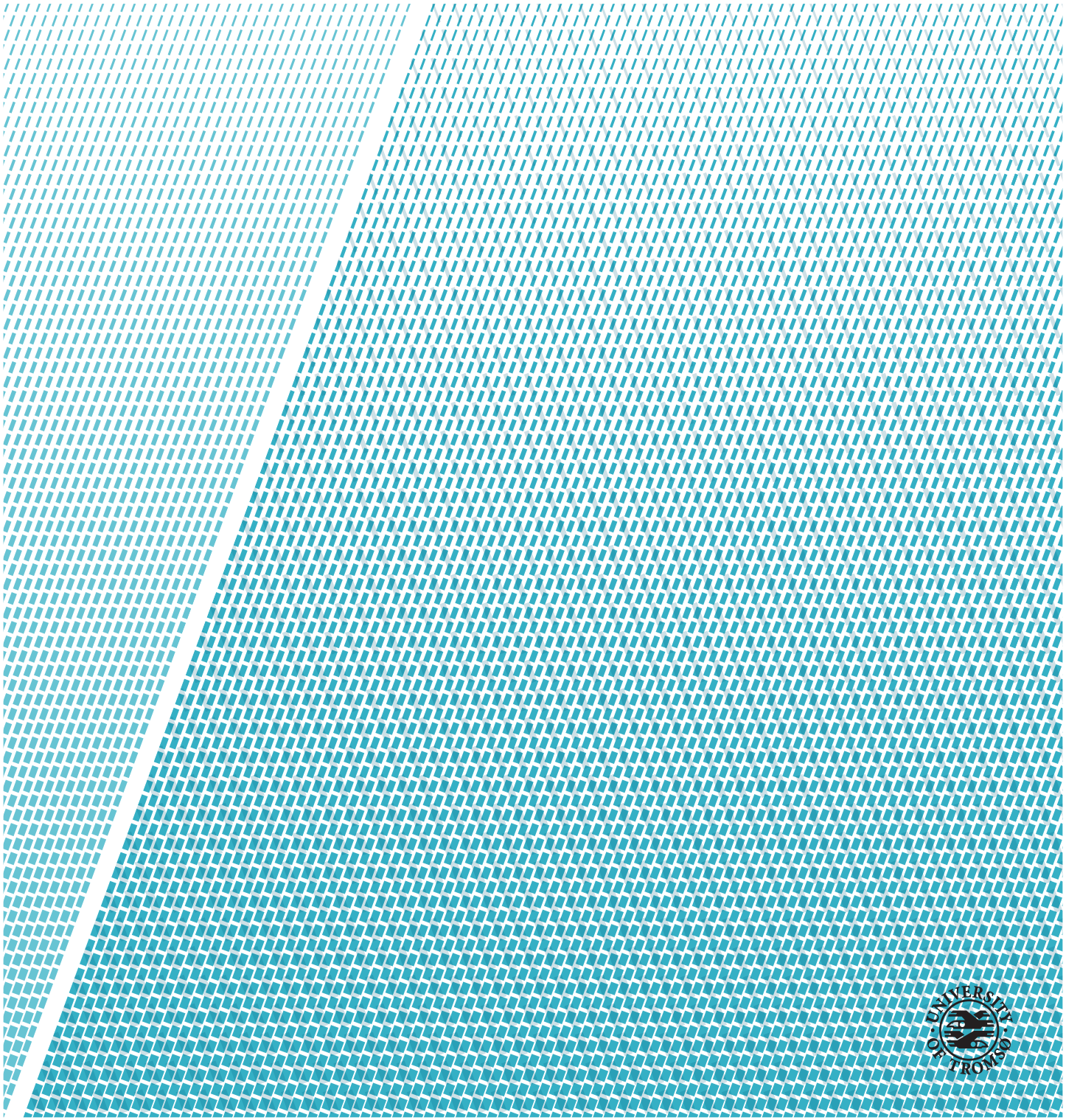
GeneNet VR: Virtual Reality for the visualization of high-dimensional relationships in bioinformatics

Subtitle

—

Álvaro Martínez Fernández

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



Contents

List of Figures

List of Tables

1

Introduction

It is very cheap nowadays to produce data and many people are doing it due to technological advancement. Just as an example, in the field of genomics, the sequencing of the first human genome (2002) took around 13 years and cost over $3 million to complete. Now we can sequence hundreds of genomes in a few days[12]. This leads to the accumulation of vast quantities of genomic data, which can be used for new scientific discoveries, diagnose of rare diseases, etc. However we still need a human expert to visually inspect the data to find new signals and discover interesting patterns or to set a diagnosis. No matter how much resources we use into extracting the data if we don't get anything interesting out of it[31]. Therefore there is a great need for new and better tools to support this task.

Some of the main problems that researchers face when analysing genomic data are information overload, data interconnectivity and high dimensionality. One way to deal with all this data is to invent novel analysis. However we still need visual inspection of the data, so the information overload still remains as an important challenge and this is what we attempt to solve. For this reason it is very important to implement efficient visualization technologies that can lead to find new patterns and the extraction of good conclusions of the data. In the field of system biology there are usually network representations where the nodes or bioentities are connected to each other, where these edges represent associations. Networks can increase dramatically in size and complexity and this is due to computational power to create very large networks, scientific knowledge about large networks and the big amounts of data that we have to

analyze. We need therefore better visualization systems for the analysis and inspection of these networks.

In Figure 1.1 we can see a representation of the evolution for visualization of networks in system biology. Before the computers, networks were represented in 2 dimensions and they were static representations that lacked interactivity, see Figure 1.1 A. With the computer era and the advancement in computer graphics, 3D representations were possible, with the addition of interactivity with the network (See Figure 1.1 C). As computer science progressed, there was a big improvement in visualization and also new technologies emerged like virtual reality, which has a huge potential with regards to the interactivity (Figure 1.1 I).

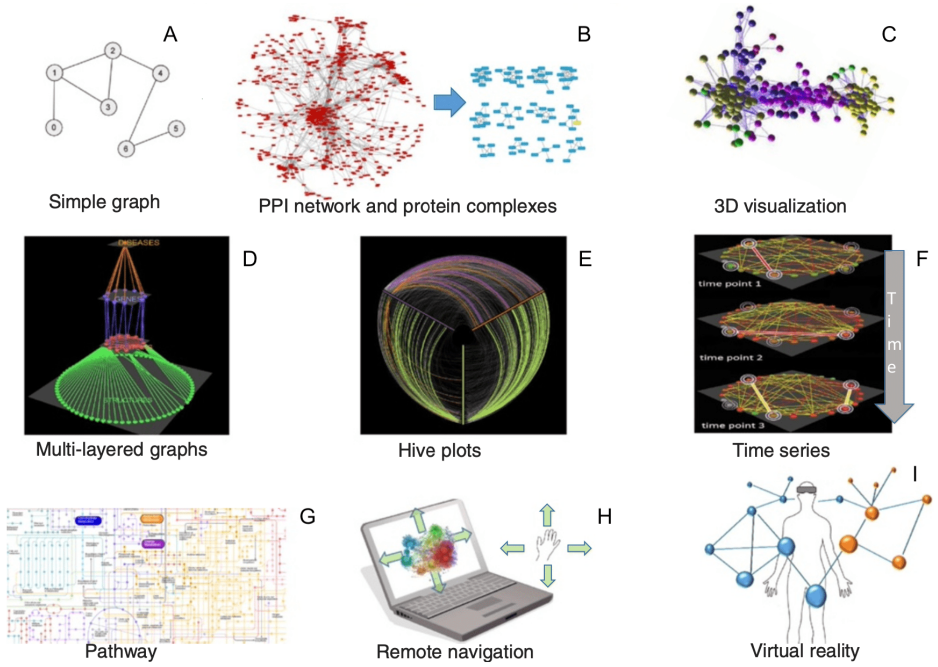


Figure 1.1: Visualization for network biology. a Undirected unweighted graph showing co-expression relationship between genes. b A 2D representation of a yeast protein-protein interaction network visualized in Cytoscape (left) and potential protein complexes 3D identified by the MCL algorithm from that network (right). c A 3D network of genes showing co-expression relationships. d A multilayered network integrating different types of data visualized by Arena3D. e A hive plot view of a network where nodes are mapped to and positioned on radially distributed linear axes. f Visualization of network changes over time. g Static picture showing part of lung cancer pathway. h Navigation of networks using hand gestures. i Integration and control of 3D networks using VR devices. Figure adapted[16].

We believe that virtual reality (VR) can offer new possibilities for visual inspec-

tion in large networks. Even though VR is still a field under exploration, it has been demonstrated that it help scientists work more effectively in fields like medicine [10][28][3], biology[29][23] and neuroscience[1][13], to cite some examples. VR can be very powerful because it takes advantage of the way the human being perceives and analysis things. We have a great ability to discover patterns, however we are biologically optimized to see the world and the patterns in 3 dimensions. Some of the advantages that VR has over non-VR approaches are the following:

1. Visualization of the network in a 3D space.

2. Possibility to move around the virtual environment and visualize the network from different perspectives as in real life.

3. Interaction with the the environment by using controllers and our virtual hands in the virtual world.

We have implemented a virtual reality application for the visualization of 3-dimensional networks of data. We have used up-to-date virtual reality techniques that we think it improves the visualization of this type of data structures. The techniques that we have used consist in: the exploration of the network by moving around the virtual space, making it easier for the user to see the network from different angles; interaction with the network and nodes to comprehend better the data that is being visualized; and the use of 2-dimensional user interfaces to filter and have more control of the data.

What did we learn... [Write when the evaluation is finished].

Thesis statment: *Virtual reality techniques can improve the visualization and interactivity of data networks.*

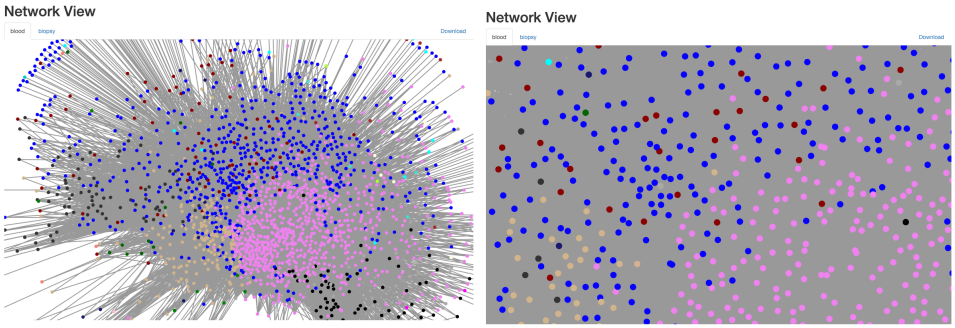
1.1 Challenges and research problem

This project focus mainly on solving the problem of visualization of high dimenasional data from the MIxT project by using virtual reality. Furthermore the application allows the user to interact with the network created from the data in the virtual environment. It also allows the user compare the blood and biopsy networks at the same time in order to finde relationship, which wasn't possible in the MIxT web application as this only allows the user to visualize one network at a time.

MIxT[8] is a web application for bioinformaticians. Among other tools, it offers

a network visualization of genes which are represented as nodes in the network and where the edges represent statistically significant correlation in expression between two nodes. This tool was used in a study[7] that identifies genes and pathways in the primary tumor that are tightly linked to genes and pathways in the systemic response of a patient with breast cancer. When exploring a network in MIxT, it can be hard to understand the data and its relationships because there is too much data. This problem is easy to occur when there are too many node and edges. In figure 1.2 we can see an example of the network visualization from MIxT. As we can see in Figure 1.2a, there are many nodes and relationships among them and when we zoom in in the network, it becomes very difficult to understand the data and the relationships as shown in in Figure 1.2b.

The network is also in 2-dimensions and what we propose in this project is to use a virtual reality 3d visualization in order to cope better with this problem.



(a) Network with several modules. **(b)** Zoom in the network.

Figure 1.2: Network view of the MIxT application where nodes repsent genes and the modules are repsented by colors. Relationships are represented by grey lines that connect a gene with another one.

1.2 Proposed solution

1.3 Significance and contribution

This project contributes in the exploration of the possibilities that Virtual Reality offers for visualization of big data in bioinformatics.

1.4 Outline

/2

BigNet 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 visible nodes using a user interface 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 cam 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 oll-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[8][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.

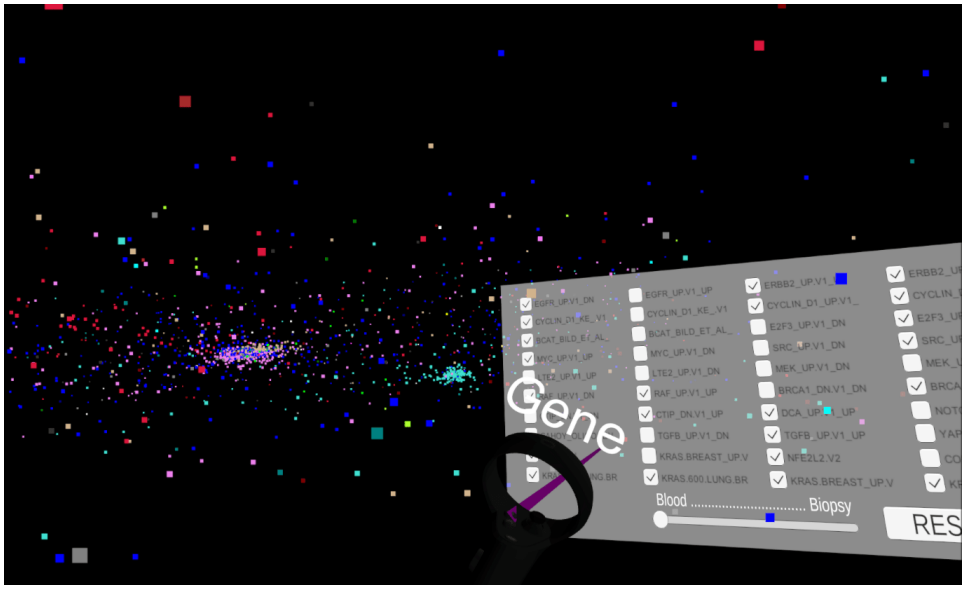


Figure 2.1: GeneNet VR. Example of the application running on a Oculus Quest.

2.1 Interaction with the network

Virtual reality headsets offer a rich inmmersive experience. It's not only about inmersing 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 interation and the visualization also depends on the VR technoly 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.

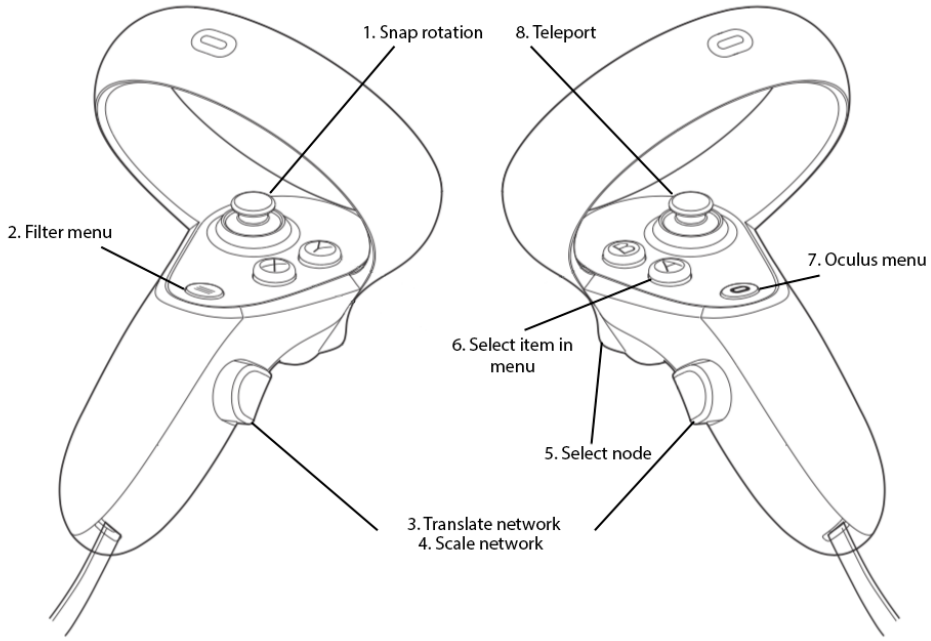


Figure 2.2: Mapping of the Oculus Quest controllers for the different actions implemented in GeneNet VR: 1. Snap rotation. 2. Filter menu. 3. Scale environment. 4. Translate environment. 5. Pointer. 6. Select item in menu. 7. Oculus menu. 8. Teleport. Adapted figure from Oculus developer's page[15].

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[2]. 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[18].

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). In addition 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.

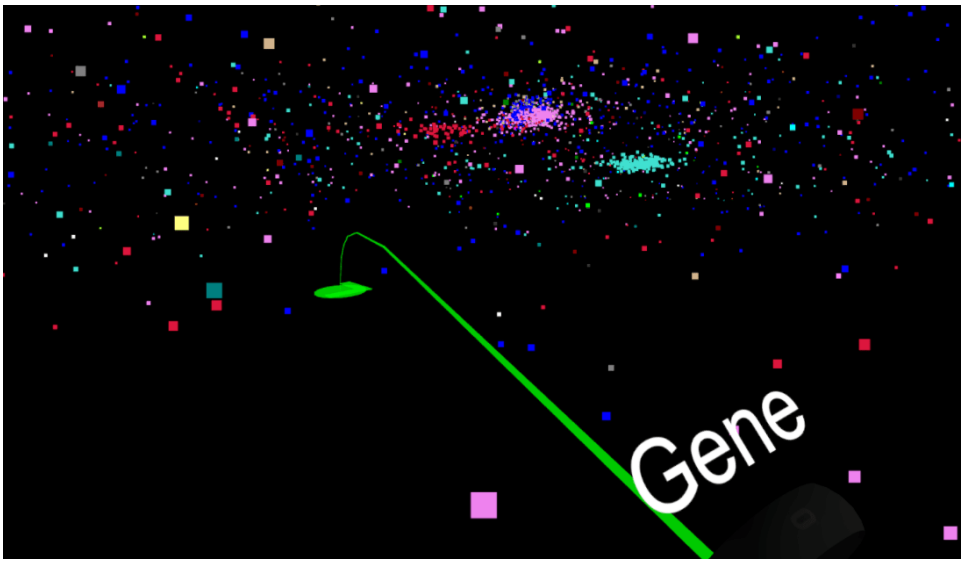


Figure 2.3: Teleportation technique. The user can use the jystick from the right controller to teleport to a different spot. To choose the spot a parabolic arc will appear.

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 thumstick 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.

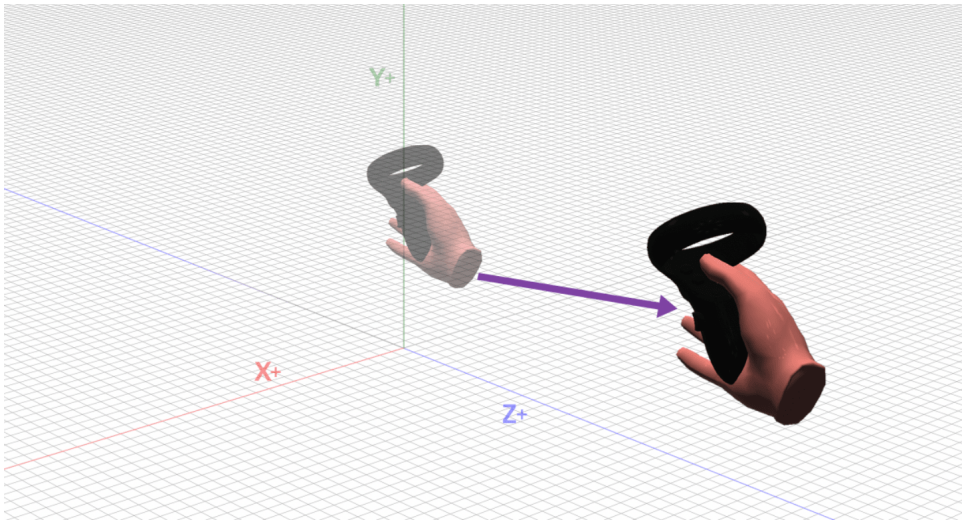


Figure 2.4: Translation of the network functionality. The user holds the translation button on the Oculus controller and moves the hand to the direction where he or she wants the network to translate.

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. 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 streching out or contracting the network itself. This is also an intuitive acction to do since the user might think that we are actually stretching the network with the hands.

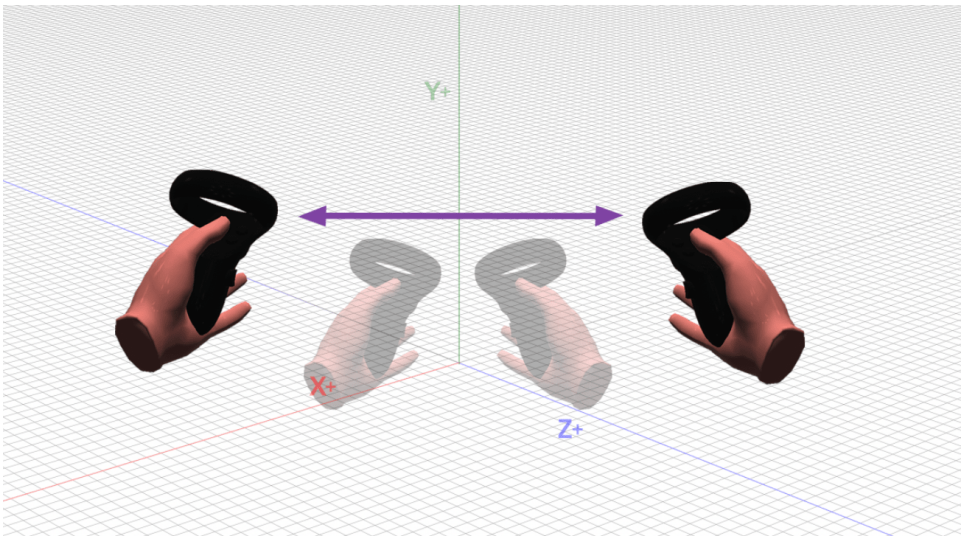


Figure 2.5: Zooming in the network functionality. The user can hold the scaling buttons on the Oculus controller to make the network bigger or smaller. In this example if we strech our hands outside, the network will expand.

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 a intial 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[4] 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 represen 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 caregory. 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[20] is defined an array of particle ojects. 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

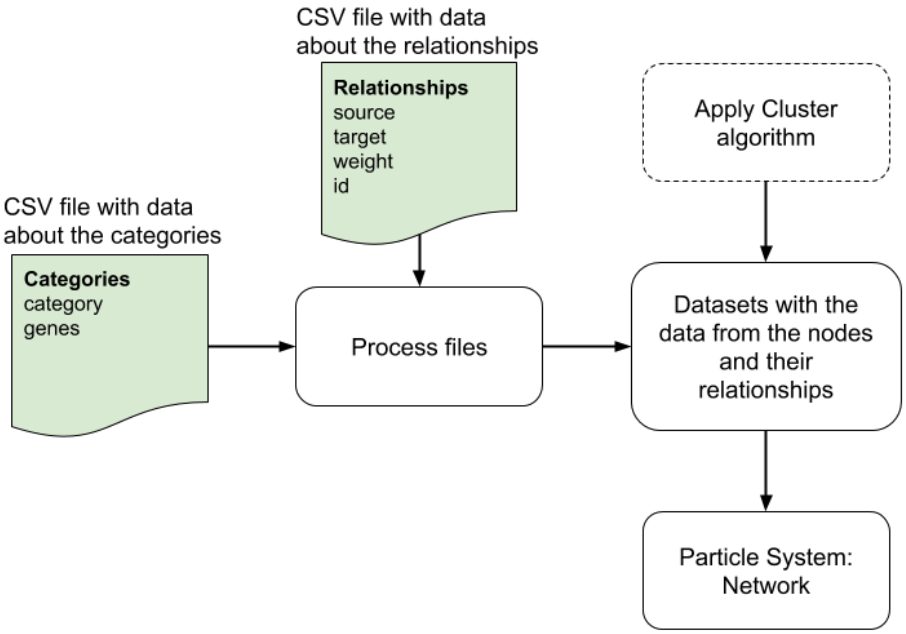


Figure 2.6: Diagram: steps for the creation of the network from the 2 CSV files.

or projectiles. They are also very powerful because they give plenty of control to the developer over the particles. In GeneNet VR we take advanatge 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. However 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 repsented with 2-dimensional lines in GeneNet VR. A line in Unity is created with a Line Render component[19]. 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 where we can which nodes are connected to each other. To store the informaion about this 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 the they are created everytime 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. Everytime an edge has to be rendered an edge object is instiantated with the CreateInstance method from Unity. The edge object in the scene from what is called a prefab in Unity. A prefab[21] 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 interesting in.

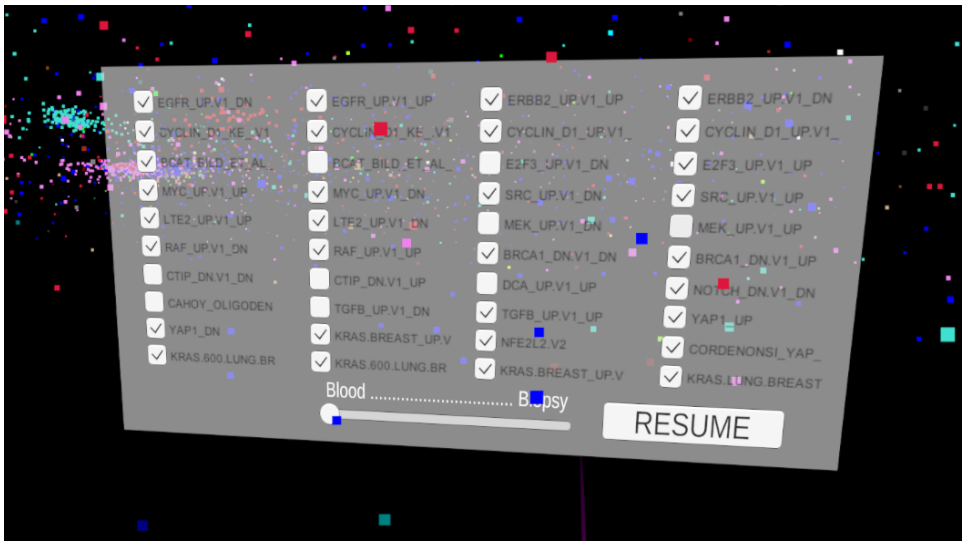


Figure 2.7: Filtering menu in GeneNet VR.

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 botton 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 botton from the right controller, named "6. Select item", see Figure 2.2.

2.3.2 Network morphing

Finally GeneNet VR has also the possibility to switch between networks. This can be done in the filtering menu by pressing the menu button from the left controller and there we can see a slider UI element as in Figure 2.7 which we can move to the right or to the left in order to switch the dataset that is being visualized. When we switch the dataset, the filters are also reset to their default state, so all of them will be checked. In our example we can switch from the blood network to the biopsy one.

2.4 Implementation details

Unity (version 2018.4.10f1[24]) 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[9]. Even though it is intuitive

to use, it also has a 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 integrations for Unity. In our case, our device is an Oculus Quest, and for this reason we use the Oculus integration for Unity[25]. In addition we have used VRTK, a collection of ascripts and assets that help build VR solutions[26]. Finally, the programming language used in Unity to implement the system is C#.

2.5 Architecture and design

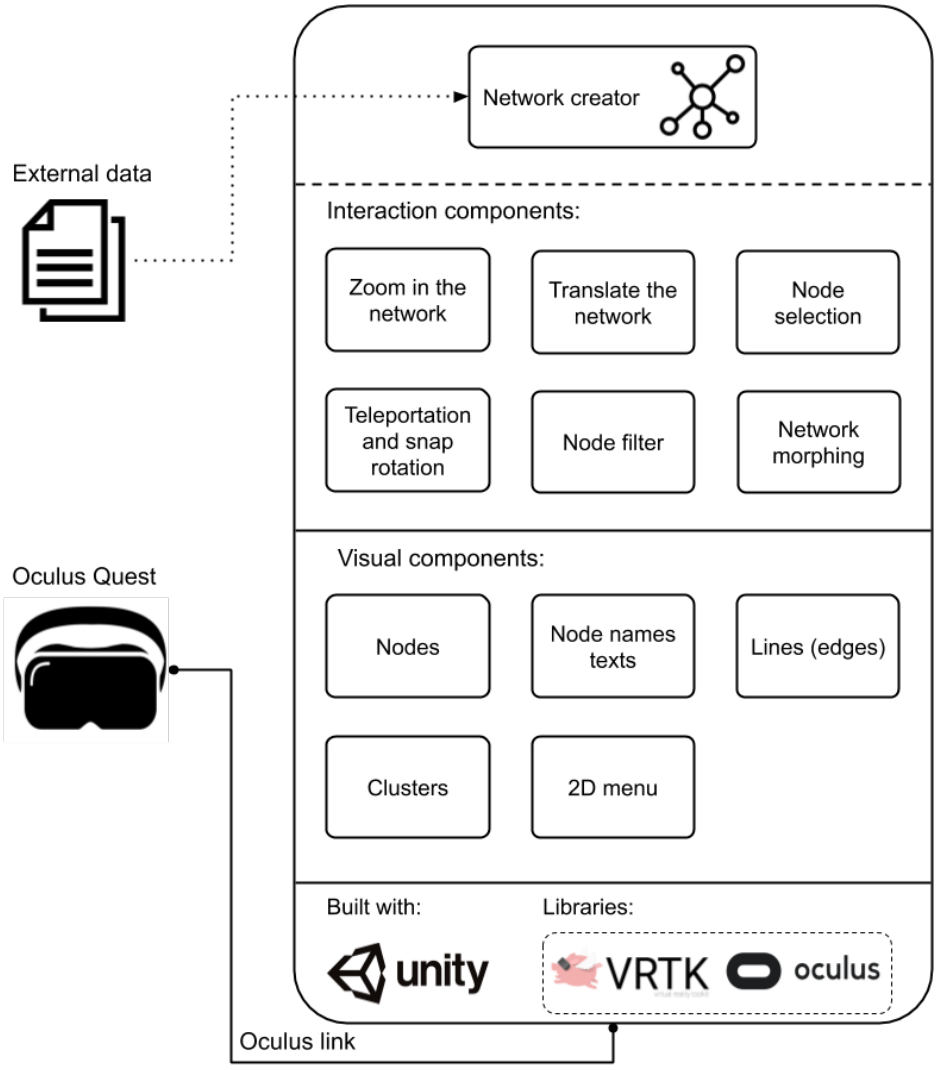


Figure 2.8: Architecture and design of GeneNet VR.

The architecture for GeneNet VR, as we can see in Figure 2.8, is divided in three parts. The main part is the application built in Unity with VRTK and Oculus libraries. Here is where we have all the logic like the network creator and the interaction features such as zooming, node selector, etc. As for the data used in the application, they come from external files and they have to be manually imported; Then they are processed when starting GeneNet VR, during the network creation. Finally there is the client, which is the Oculus Quest headset where we run GeneNet VR. Here the user can visualize the network and use the controllers to interact with it, using the interaction features that we have created. As the figure shows, the Oculus Quest can be connected to the PC itself using an Oculus Link which is basically a high-quality USB 3 C to C or USB A to C cable with proven performance[14]. This allows the user run GeneNet VR on the PC. Another possiblity is also load GeneNet VR into the Oculus Quest and run it in the headset hardware without any cable or PC. We are going to explain now in more detail how each of the elements of GeneNet VR were implemented. In this subsection, the actions that are triggered using the Oculus controllers are specified in Figure 2.2.

The network creator (see Figure 2.9) is the component of the application that builds the network from the external data. 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. During this process of building the network we apply the clustering algorithm as well.

For the *zoom in the network* component I wrote a C# script where I use the Oculus integration in order to communicate with the controllers of the Oculus Quest. The algorithm is run everytime the user enables the action for zooming (see Figure 2.10). 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 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 one is smaller than the initial one, the network will srink; if it is bigger the network will grow up.

For the *translate the network* component I also wrote a script in C# and using the Oculus integration. Here we do something similar to the *zoom in the network* component. Right before the action 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 at every frame. Then a vector is calculated like (current_position - initial_position) and normilized. With this we obtained the direction of the

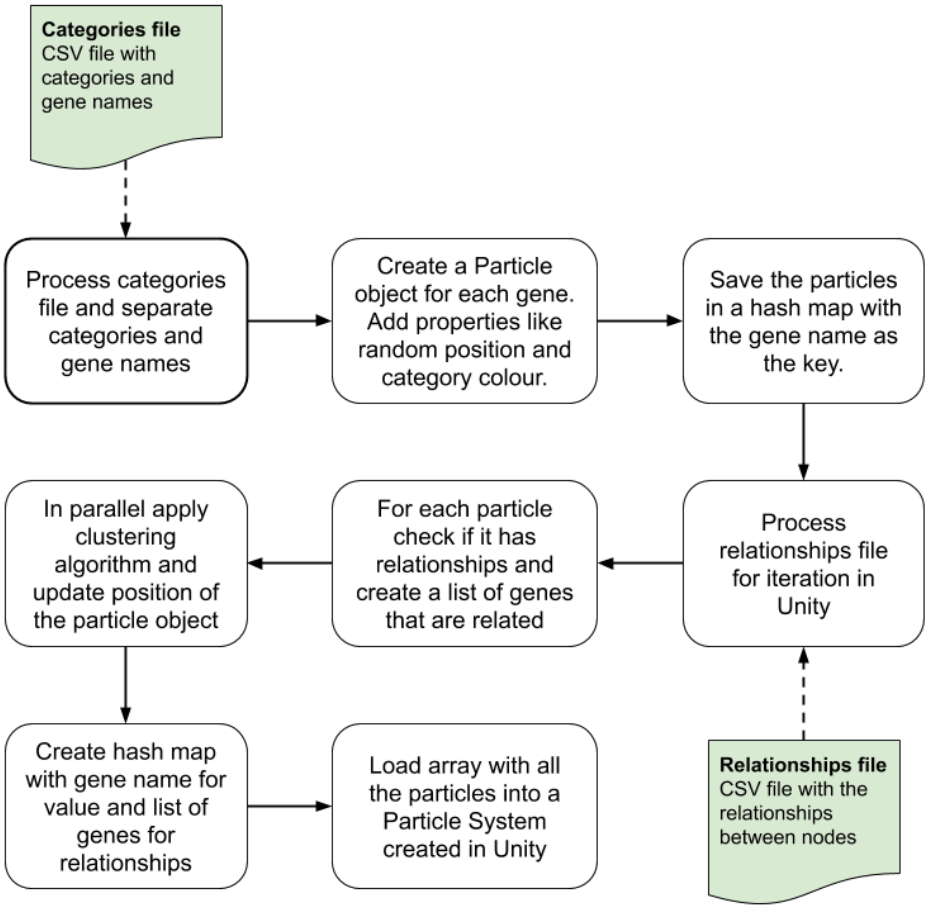


Figure 2.9: Network creator algorithm.

vector where the user is trying to move the network to. Then we just add up that vector with a constant to update the position of the network in the 3D space.

Select node algorithm:

For the node filter component, a UI interface is used in Unity. This interface contains several checboxes that are switched on by default. In addition, each checkbox has attached a function that is run everytime the user switches it off or on. This function receives a a text variable as a parameter that is used for the filtering. When the user switches off the checkbox, the algorithm looks into a hash map that looks like this:

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

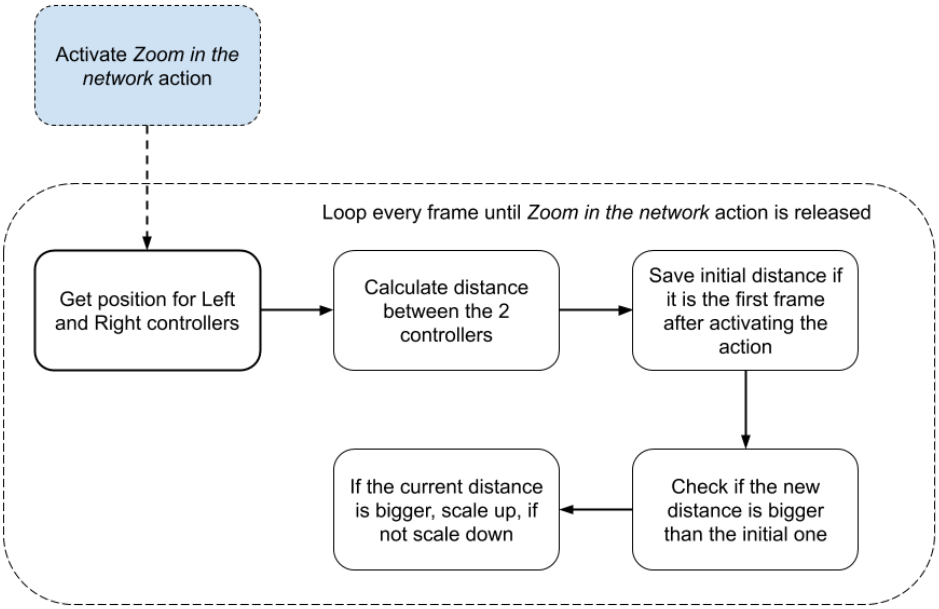


Figure 2.10: Algorithm for zooming in the network.

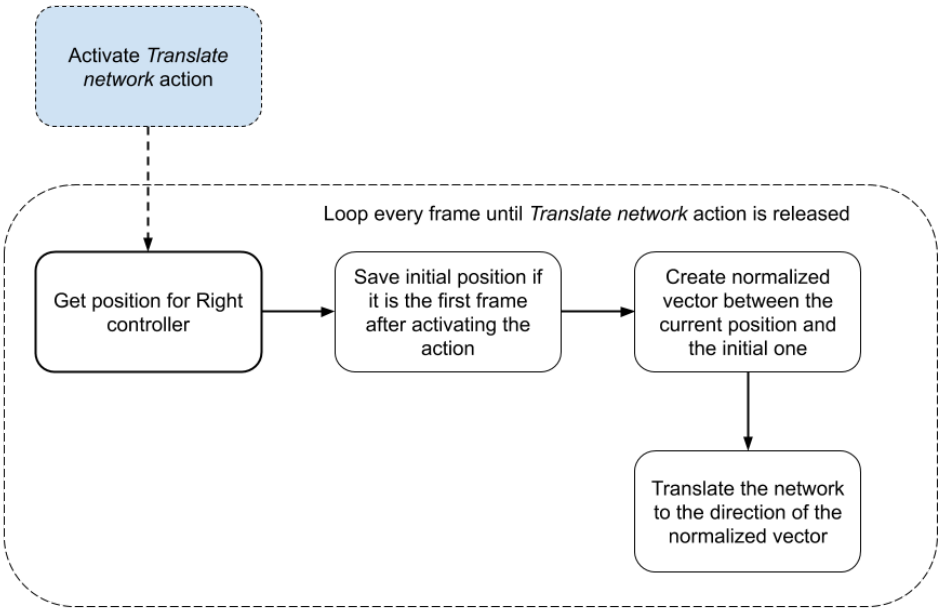


Figure 2.11: Algorithm for translating the network.

We use the string variable that is passed to the function and that is used 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 off from

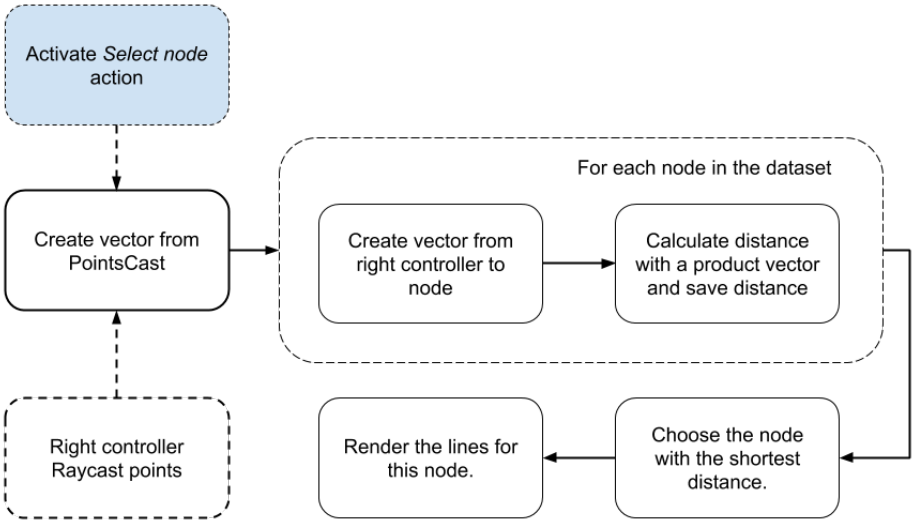


Figure 2.12: Algorithm for the selecton of the nodes in the scene.

the network.

2.6 Discussion

2.6.1 Unity vs Unreal Engine

Unity3D[1] and Unreal Engine[2] are two popular softwares for the development of videogames as well as virtual reality games and other applications. They offer integrations 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 about VR development, I found that Unity is good 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 and they conclude that Unity is little less complicated to learn[6]. On the other hand Unreal can create more professional looking results. However we are not looking to have realistic graphics in our application since we are dealing with abstract data.

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

2.6.2 VR libraries

2.6.3 VR headsets

We can find numerous VR headsets nowadays in the market[3], 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 the features that the headset has. We chose Oculus Quest because the price is not very high (5799,- kr in 2019) and it was also an 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 the 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 buttons, triggers and grips that can be used for the interactions in the VR application.

3. https://versus.com/en/vr-headset

3

Visualizing MIxT in VR

What is MIxT used for? And how would users do it in VR? How the MIxT application is implemented in GeneNet VR and what it does. What are typical network characteristics for this (type of) application? Size, clusters, edges, connectivity, etc?

4

Related work

I will focuse in this chapter on VR applications found in the literature for the visualization of bioinformatic data.

4.1 Virtual Reality Chemical Space

Virtual Reality Chemical Space is a VR application for the interactive exploartion of chemical space populated by Drugbank compounds[17]. 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 the 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 4.1 we can see an 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 the 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 analogue 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 conlcude that the application of VR in chemistry has more potential in the fields of education and training as for the current state of technology. The tools for chemistry need to be further evaluated wether they should be extended to VR.

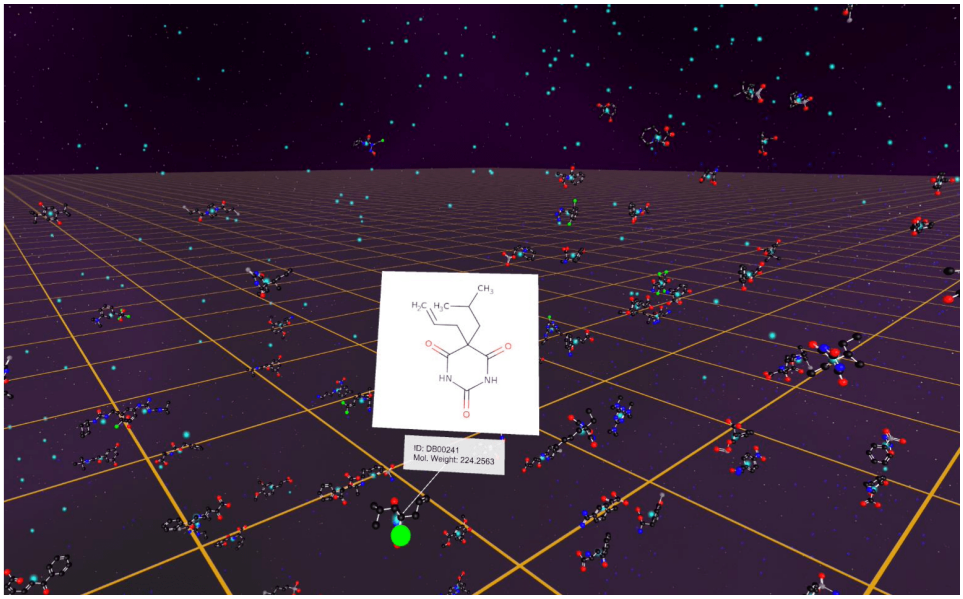


Figure 4.1: Optimized virtual reality chemical space. Figure taken from [17].

4.2 BioVR

BioVR is an interactive VR platform for integrated visual analysis of DNA/RNA protein structures[30]. 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 in Oculus Quest, making it easier for dveleopment[1]. An screenshot of BioVR can be seen in Figure 4.2. 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. They also of VR is that it is easy to integrate 2D user interfaces in the virtual world, as we can see in Figure 4.2 B.

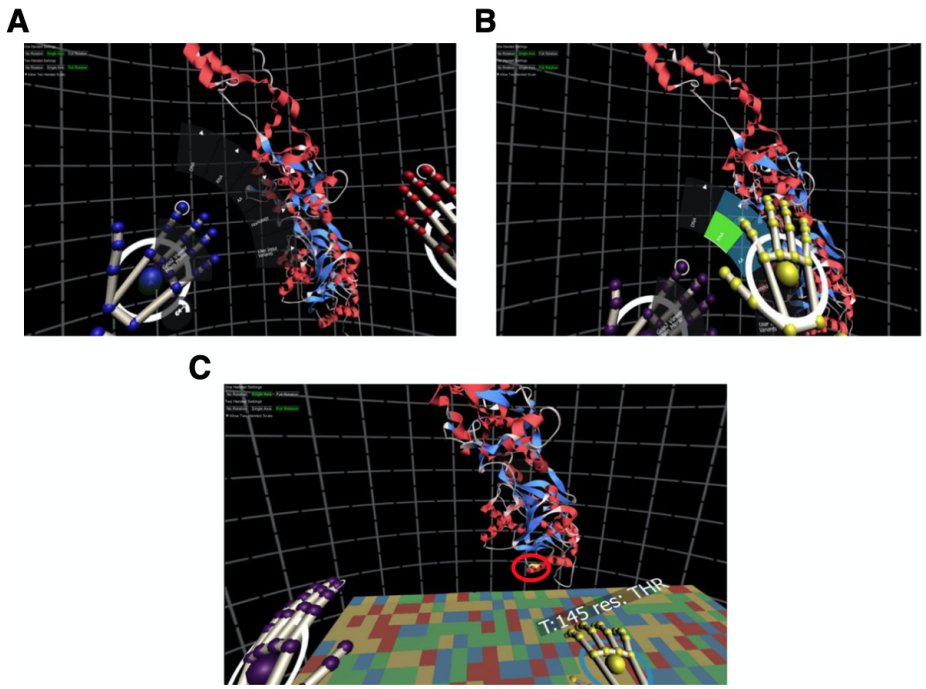


Figure 4.2: Screenshot from BioVR. Figure taken from [30].

4.3 CellexalVR

CellexalVR is a virtual reality envirnoment for the visualization and analysis of single-cell RNAseq experiments that help researchers undertand their data[11]. The system is divided in two parts: the first one consists on 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 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 implemenation.

Something that is interesting in CellexalVR is that it allows a multi-user mode via the Photon Unity Networking. This works sending information about the events to each user using Remote Procedure Calls. In Figure 4.3 we can see an 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. Espcially the "ghost mode" could be useful so that other people can visualize the network at the same time from other perspectives.

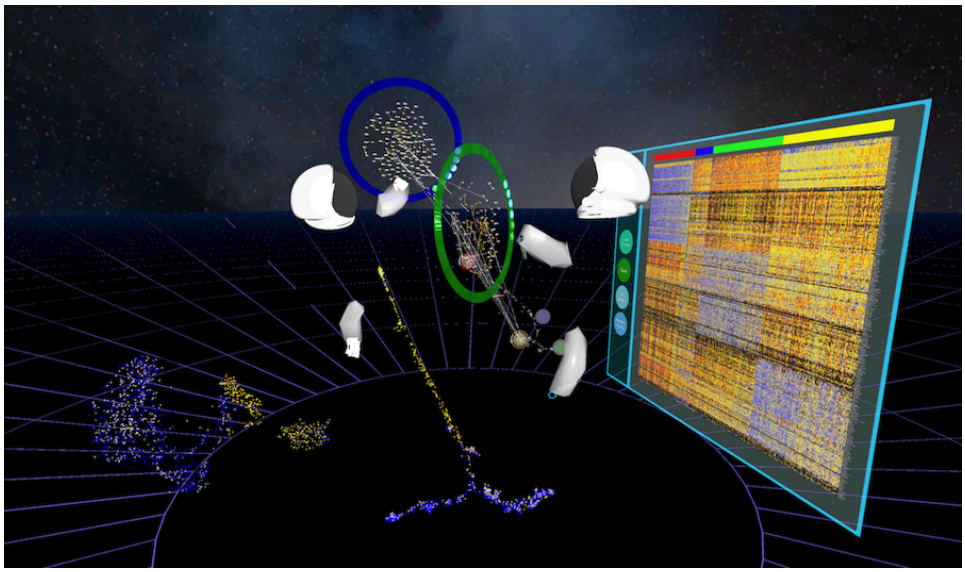


Figure 4.3: Screenshot from CellexalVR. Two users using CellexalVR at the same time. The head models were taken from NASA. Figure taken from [11].

4.4 BigTop

BigTop is a visualization framework in VR for the rendering of Manhattan plots in three dimensions[27]. Manhattan plots are usually 2-dimensionals, where genomic coordinates are displayed in 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 a SNP then. In BigTop, the z-axis is used to display 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 4.4 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 isong click and drag with the mouse (in the browser). In GeneNet VR we have focused only into building a visualization system for a VR headset. We have used better locomotion techniques that improves the comfortability. However the locomotion technique that they use to move around could be a good idea for GeneNet VR as well.

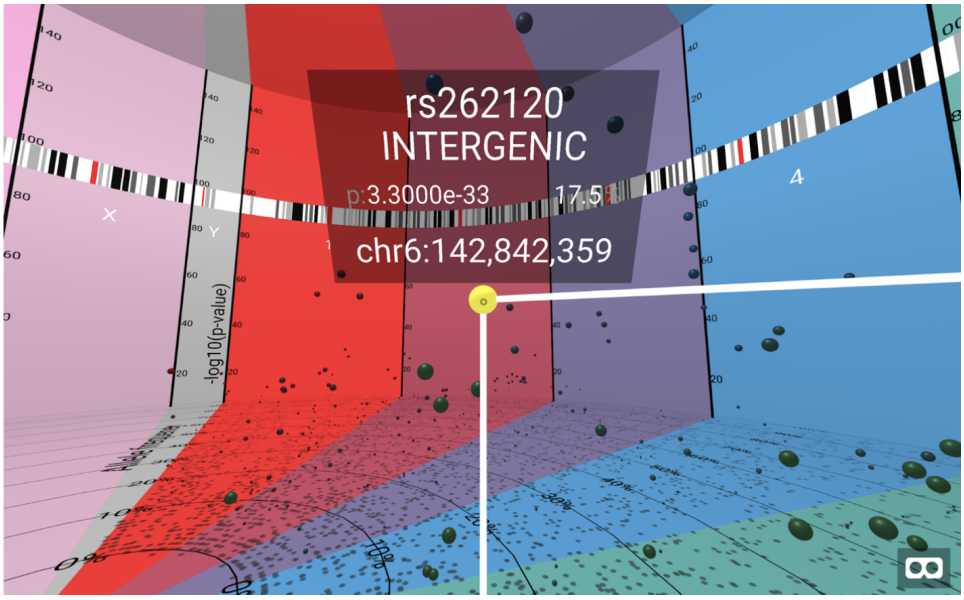


Figure 4.4: Screenshot from BigTop where a node is selected. Figure taken from [27].

5

Evaluation and discussion

GeneNet VR has been built to explore biological networks that contain genetic information from human beings. We have used two datasets from MIxT where we applied several VR techniques to build a visualization system. In this chapter we want to evaluate this system focusing on scalability and performance problems. Since we are visualizing networks with genetic information using VR, we want to know if the system that we built can be used for any size of this type of network. As part of the evaluation process, we designed a list of questions that we will try to answer along this chapter. The questions are:

1. For which interactions do we achieve the recommended FPS (72) when scaling the network?

2. What characteristics of the network influence the scalability?

3. What is the performance by using Oculus Link and the performance using just the Quest hardware?

4. How do users perceive the interaction of the network?

Question one is based on the Oculus' performance guidelines[22], that say that an application should meet 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.

As for the second question, we want to know what characteristics influence the scalability of the system. Some parts will have more impact in the performance than others. To keep it simple, we will evaluate the impact in the performance done by the following elements, that are the most important for the visualization: number of nodes, number of lines and number of clusters.

We will also evaluate the performance of the application being run in the Oculus Quest hardware and also in a PC, using the Oculus Link cable that connects the headset to the machine. The hardware of the Oculus Quest is not as powerful as the one of the machine that was used for the development. We would like to know if the performance is good in both the machine and the headset.

Finally, as the last question says, we want to know how the users perceive the interaction and visualization of the network. We will evaluate this with a qualitative method with a demo of the application using the MIxT datasets and also an interview.

5.1 Experimentation plan

An experimentation plan was designed to ensure that the experiments are consistent and that they can be repeated several times getting realistic measurements. We take into consideration the following aspects for our experiments:

1. Scalability for different interactions.
2. Network characteristics.
3. Bottlenecks.
4. User study.
5. Hardware and software specification.

There are some elements of the network that influence more in the scalability and performance of the system. In Table /reftab:network-elements we describe these elements.

Element	Description
Clusters	description for clusters
Nodes	Represented as 2D squares in the space. The position is calculated when creating the clusters.
Lines	They represent relationships between the nodes. Everytime a node is selected, line objects are added to the scene.

Table 5.1: Elements of the network that have influence in the scalability.

We ran the experiments in a machine with Windows 10. In Table 5.2 we can see some of the hardware specification from the machine used:

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

Table 5.2: Machine specification.

Since the GPU is important in this type of applications, we show the specs of the GPU used in Table 5.3.

GPU specification	
Adapter type	NVIDIA GeForce GTX 1080 Ti
Chip Type	GeForce GTX 1080 Ti
DAC Type	Integrated RAMDAC
Available memory	45025 MB

Table 5.3: GPU specification.

As for the VR headset hardware, we used a Oculus Quest. In Table 5.4 we can see the hardware specifications for this type of headset.

The experiments were run in Unity, the same software that was used for the development. The version of Unity used for the expriments is 2018.4.10f1. Also Vulkan is used as the graphic API.

A test scene was built in order to test the network and the scalability of it. A random network is built and we test how comfortable it is to navigate through it. This is measured by the FPS rate.

Oculus Quest specifications	
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 5.4: Oculus Quest specifications.

VR profiling is a technique used to get an overview of the performance of our application. This is usually done in order to find bottlenecks so that we can eliminate them and improve the application's performance.

To profile the application we used the built-in profiler in Unity, the software used for the development. The Unity Profiler gives information about per-fram CPU and GPU performance metrics.

5.2 Performance and scalability of the system

We have run some experiments where we analyse the scalability of the system. We decided to analyse the parts of the system that can have more impact in the scalability and that are most used by the user when exploring the networks. These are: moving the network, scaling the network and node selection (which also includes the cration of relationships in the scene).

I will describe now the experiments that I want to run to evaluate this. The experiments will be divided in two parts. The first part consists of several experiments where I analyze the performance for each of the actions that I will focus on. The second part will focus on analyzing the scalability for the line creation, which can create bottlenecks, according to the performance analysis done.

5.2.1 First part: performance

In the first set of experiments, we are going to use the profiler from Unity, where we can see a frame chart that contains information about performance data over time on a frame-by-frame basis[1]. We will use the blood dataset from MIxT for these experiments and we will manipulate the size of it for each of the

1. https://docs.unity3d.com/Manual/ProfilerWindow.html

actions that we want to evaluate: one version with a third of the nodes from the dataset, one with half of the nodes and then a final one with all the nodes. We will also remove the relationships for the nodes that doesn't exist anymore from the reduced datasets. In total we will run 9 experiments (3x3). This will produce a total of 9 screenshots that I could have as anexus (A page can contain 3 of these screenshots, each page for a different action). Furthermore I will resume the data in 3 tables like Table 5.5, Table 5.6 and Table 5.7. Each table shows the data from the experiments for a specific action. The last column shows the average of the 25% of the worst times (so the 25% of the frames that have higher values). An alternative to the tables could be a bar graph.

As for how I will run the experiments, I want to create a script for each action. I want to create a script that moves the network, another script that scales up and down the network and a third one that selects a few nodes. We can select the 100 first frames an run the calculations for the percentages. This can be a tedious task if I have to do it manually, it would be better if I can do it programmatically (which I haven't seen a solution for so far).

I run the experiments three times and calculate the average. For translate and scale experiments I calculate the time frame for a range of 1000 frames, from frame 501 to frame 1500 inclusive.

700 frames in total for all the experiments.

Dataset size	1% average low	0.25% average low	Average
size	9.16	12.531	7.205
size/2	8.625	11.094	7.215
size/3	8.767	11.619	7.068

Table 5.5: Performance of translate the network.

Dataset size	1% average low	0.25% average low	Average
size	9.195	11.955	7.263
size/2	8.856	11.608	7.125
size/3	8.933	11.8	7.112

Table 5.6: Performance of scale the network.

The blood dataset has a total of 2693 nodes. Each node can have between 10 and thousands of edges with other nodes. The node with the name ARGLU1 has the highest number of edges, a total of 1607. In Figure 5.1 we show a scatter plot where the X axis represents the the number of edges in ascendent order and the Y axis represents the number of nodes that have that number of edges. As we can see, most of the nodes have less than 200 edges. But a few ones have several thousands. We have selected a few nodes for the experiments that

cover nodes with lower number of edges to higher number of edges. The nodes are: TGFBR3 (1), EPSTI1(11), SMNDC1(90), HNRNPH3(290), ANGEL2(586), ACTR6(756), FOPNL(1145), ARGLU1(1607).

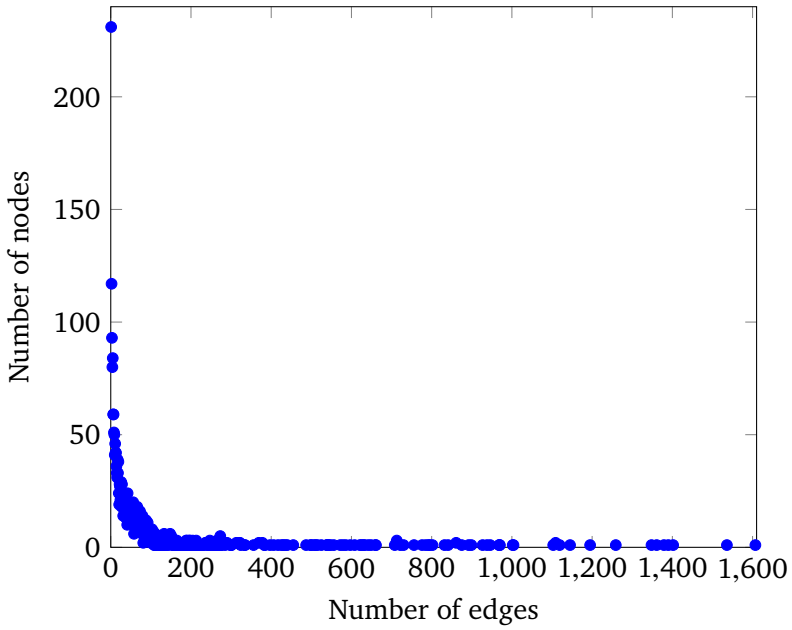


Figure 5.1: Scatter plot showing a distribution of the number of edges in the blood dataset.

Dataset size	1% average low	0.25% average low	Average
size	16.911	35.822	9.113
size/2	15.594	33.333	8.647
size/3	14.266	29.82	8.388

Table 5.7: Performance of select node.

5.2.2 Second part: scalability

Once we see that the selecion of the nodes (including drawing the lines) has more impact in the performance and that it can create bottlenecks, we can evaluate the scalability of the line drawing. For this I can create an experiment where I measure the time that it takes to draw the lines for the selection of 15 nodes (20 can be too many for the graph). I can do three versions of this experiment where I scale the number of relationships by 2 and 3 times. I can also show the number of lines that are drawn for each node.

I can create a linear graph for this, like the one I showed (See Figure **??**). I would also like to show the number of lines that each node renders. Maybe I

can show this in the x axis. Any suggestion?

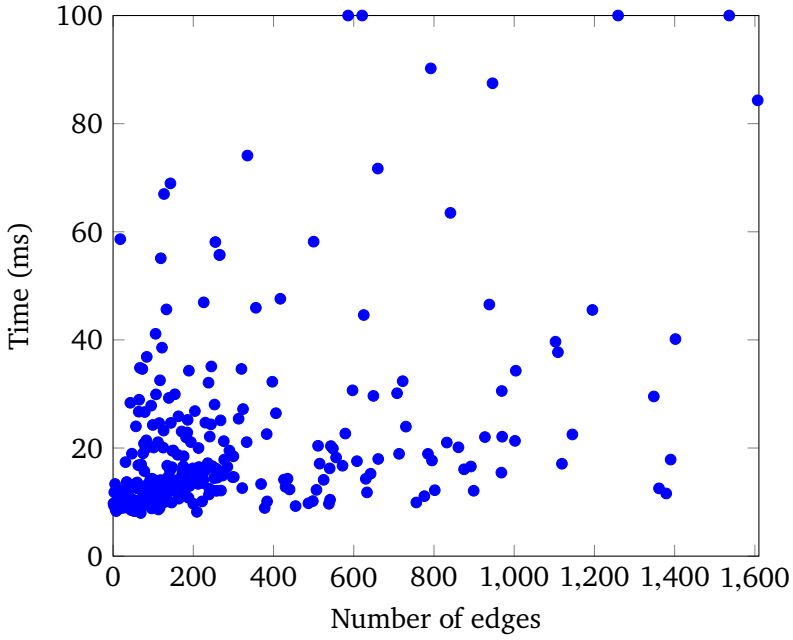


Figure 5.2: Scatter plot showing the relation between the number of edges to render and the time that it takes to render for the blood dataset.

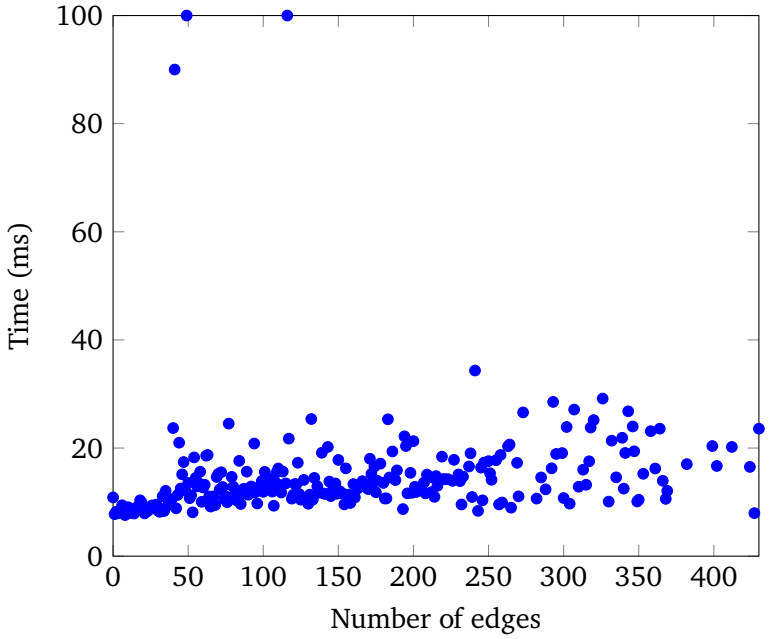


Figure 5.3: Scatter plot showing the relation between the number of edges to render and the time that it takes to render for the biopsy dataset.

5.3 Oculus Quest vs Computer

Create an experiment where serveral actions are run at the same time: translate, scale and select nodes. Capture the average low for several nodes.

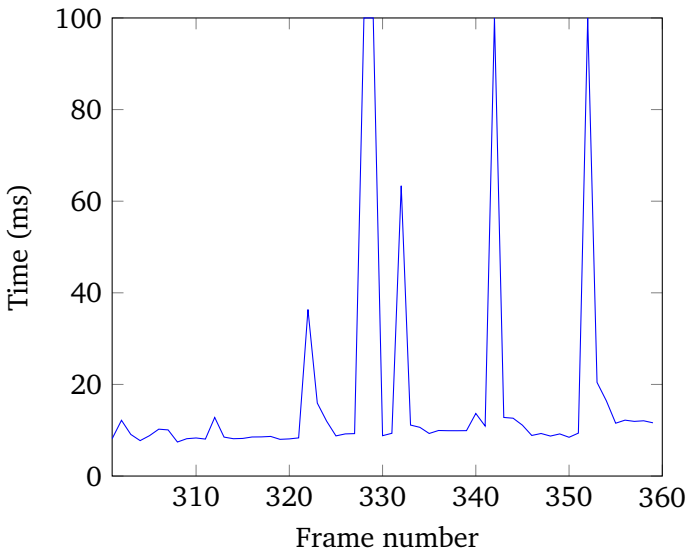


Figure 5.4: Performance running the application in a machine vs on Oculus Quest.

5.4 Demo and interview

One of the questions that we asked ourselves during the evaluation process was about the comfortability of using BigNext VR to explore a biological network. This is an important aspect when building VR applications. Some of the aspects to take into account are for instance the motion sickness or the intuitiveness. In order to evaluate this we made a questionnaire for bioinformaticians that would test the application. Unfortunately, due to the Covid-19 situtation[5], we were not able to carry out the questionnaire with people. The reason is because it wasn't possible to test GeneNet VR with people on a single Oculus Quest device without avoiding the social distancing rules. We estimated to have around 10 participants with knowdledge in bioinformatics to test the application. With this number of participants we could have made some statistics and obtained feedback for future improvement.

The following questionnaire is divided in four sections; a general section about VR headsets, a section about comfortability exploring the network using GeneNet VR, a section about the different actions in GeneNet VR and finally a section about feedback.

To complete the questionnaire, the teste has to indicate the level of agreement or disagreement with each of the statements, mark yes or no when it is asked and in the feedback section reply the questions with constructive feedback if possible.

6

Conclusion and future work

In this section we will conclude the thesis and describe what we can do to improve the project and the future work.

6.1 Future work

Bibliography

[1] 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.

[2] 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.

[3] "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.

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

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

[6] 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.

[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] 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.

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

[10] 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.

[11] 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.

[12] 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.

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

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

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

[16] Georgios A. Pavlopoulos et al. "Visualizing genome and systems biology: technologies, tools, implementation techniques and trends, past, present and future." In: *GigaScience* 4.1 (2015). DOI: 10.1186/s13742-015-0077-2.

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

[18] 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.

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

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

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

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

[23] 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.

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

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

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

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

[28] 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.

[29] 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.

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

[31] 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.