Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Fakultät für Mathematik und Informatik
Institut für Informatik
Games Engineering

Bachelorarbeit

zur Erlangung des Grades
Bachelor of Science (B.Sc.)
im Studiengang Games Engineering
an der Universität Würzburg

Interactive Visualization and Comparison of Graphs in Virtual Reality

vorgelegt von
Michael Kreuzer
Matrikelnummer: 2095906

am 10.05.2019

Betreuer/Prüfer:
Prof. Dr. Sebastian von Mammen, Informatik IX, Universität Würzburg
Prof. Dr. Alexander Wolff, Informatik I, Universität Würzburg

Danke an Prof. Dr. Juliano Sarmento Cabral und Ludwig Leidinger
für die Bereitstellung von Datensätzen und Feedback zur Anwendung

Abstract

As virtual reality hardware becomes more mature and affordable its use in information visualization is getting more common. We explore possibilities for the visualization of graphs in an virtual reality environment with focus on intuitive interaction, automatic layout generation and dynamic visualization of additional vertex attributes. Furthermore we analyse methods of creating a comparison view between two graphs with common vertices in this virtual reality environment. We show an exemplary use case of such a visualization with the use of sample data from bioecological coexistence simulations

Zusammenfassung

Da Virtual-Reality-Hardware immer ausgereifter und erschwinglicher wird, wird auch ihr Einsatz in der Informationsvisualisierung immer häufiger. Diese Bachelorarbeit untersucht die Möglichkeiten der Visualisierung und Interaktion mit Graphen in einer Virtual-Reality-Umgebung. Hierbei liegt der Fokus auf intuitiver Interaktion, automatischer Layoutgenerierung und dynamischer Visualisierung zusätzlicher Knotenattribute. Darüber hinaus werden Methoden zur Erstellung einer Vergleichsansicht zwischen zwei Graphen mit gemeinsamen Knoten in dieser Virtual-Reality-Umgebung analysiert. Wir zeigen einen exemplarischen Anwendungsfall einer solchen Visualisierung unter Verwendung von Beispieldaten aus bioökologischen Koexistenzsimulationen.

Contents

1 Introduction

Virtual and augmented reality are concepts that introduce computer generated elements into the environment. While augmented reality aims at enhancing the perception of the real world by displaying additional computer generated information about real world objects in the users field of view, virtual reality (VR) replaces the whole perceived environment with an artificial computer-rendered world. This is usually done by using head-mounted displays that cover the entire field of view of the user.

With VR, the user can directly interact with the scene he or she is presented with by touching or grabbing objects with just his or her hands instead of having to use some kind of pointing device. Furthermore the user can be placed inside the visualization with elements all around him/her instead of just looking at a computer monitor.

While ideas and concepts for virtual and augmented reality have been around for a long time, up until a few years ago, computer hardware was just not able to deliver enough performance for this technique. But as hardware has gotten better, a number of head-mounted displays (HMDs) have been released in recent years, two of the most popular being the HTC Vive[1] and the Oculus Rift[2]. These HMDs feature accurate positional tracking and high-resolution displays that are placed directly in front of the users eyes. Many HMDs ship with specifically designed controllers that allow tracking of the user's hands as well. Through the availability of these affordable consumer-grade HMDs the use of VR not only in games but also in industry applications such as complex visualizations is as common as never before.

The visualization of graphs is a widespread topic in research focusing on how to draw different forms of graphs in an aesthetically pleasing way. Up until now, most of the work in this area focuses on 2D drawings, although there is some research done on

[1] https://www.vive.com/eu/product/
[2] https://www.oculus.com/rift/

3D visualizations as well. Large graphs still pose a great challenge in visualization and are often just too complex for 2D visualizations. This provides great potential for VR applications, which allow for an actual immersive 3D visualization without being constrained to a 2D monitor of limited size. Chapter 2 goes into more detail on similar approaches and lists related work of this thesis.

Large graphs are a common occurrence in bioecology for the visualization of ecological simulations that often consist of many different agents. Examples of such models are described by Cabral, Valente and Hartig (2016). Therefore we have developed an application of which a picture is shown in Figure 1.1 to display the coexistence of species in a mechanistic, eco-evolutionary simulation model. The application is an interactive visualization of two similar scenarios that should be compared. Each scenario is represented by a graph with vertices corresponding to species and edges corresponding to the strength of coexistence between two species. The user can interact with and move around each individual vertex but also change the appearance of the whole graph by selecting different visualizable attributes. An comprehensive overview over all features is provided in Chapter 3.

The application has undergone an iterative development and evaluation process, with multiple testing and feedback sessions performed together with two bioecological researchers. This process and the evaluation results are presented in Chapter 4. The project is summarized in Chapter 5 which also provides ideas for future work that could be performed based on this thesis.

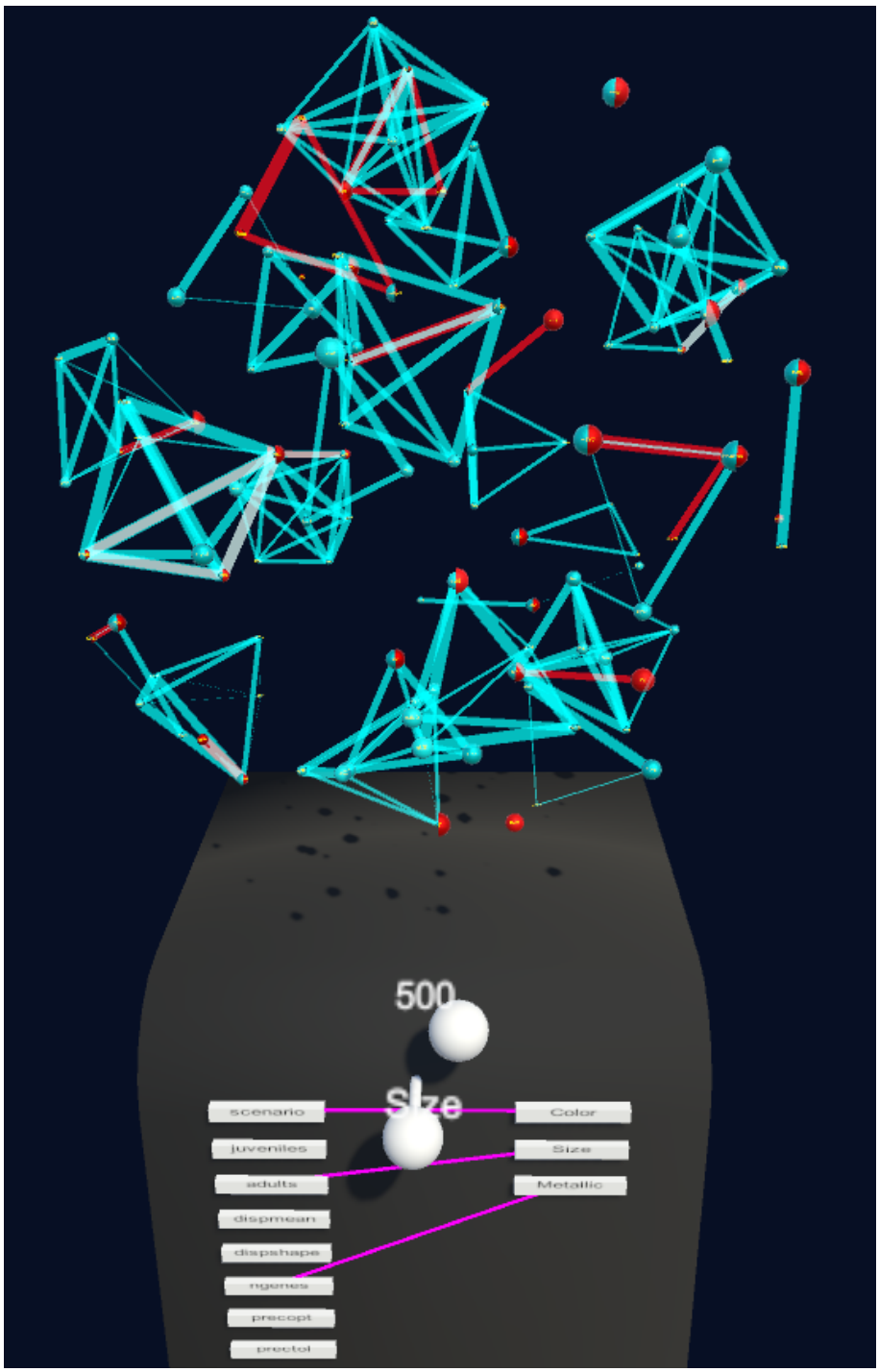

Figure 1.1. An overview over the application. The central element is the graph that is coloured in blue and red depending on the scenario each vertex and edge corresponds to. The user can freely move around vertices and interact with the sliders and connection buttons that are mounted on a table below the visualization. A detailed explanation of this UI is given in Section 3.3

2 Related Work

Although VR is a relatively new technology (at least in practical use), the use of VR in the field of graph visualization has already been investigated. Nevertheless, this bachelor thesis uses concepts from non-VR fields of research as well, which will be detailed in the following sections.

2.1 VR Graph Visualization and Interaction

As of today there are only very few scientific publications that use commonly available consumer virtual reality hardware such as the HTC Vive[1] or the Oculus Rift[2] for interacting with graphs in VR. Nevertheless Huang, Fujiwara, Lin, Lin and Ma (2017) and Erra, Malandrino and Pepe (2018) present possibilities of interaction with 3D graphs using the Leap Motion system, a hand and finger tracking device.[3] They both programmed gestures for moving around vertices and edges, rotating and translating the whole graph and grouping or ungrouping of vertices. Both papers conduct studies on the effectiveness of their visualizations. Huang et al. (2017) find out that interaction with gestures recorded by leap motion performs better compared to mouse interaction in a VR context. However the results by Erra et al. (2018), who compared an Oculus/LeapMotion combination with a traditional screen/mouse combination, show that VR does not necessarily perform better. This might be caused by some limitations on the virtual reality setup of the study such as a seating position which does not allow for a lot of natural head movement and requires additional locomotion techniques (apart from the natural head movement of the user) which are prone to causing motion sickness. Despite these limitations most participants in their study found the VR visualization more enjoying and satisfying than the non-VR counterpart. Millais, Jones and Kelly (2018) found similar

[1] https://www.vive.com/eu/product/
[2] https://www.oculus.com/rift/
[3] https://www.leapmotion.com/

evidence in their study on 2D versus VR interaction, but mention that VR produced fewer inaccurate data insights.

Continuing on the topic of interaction, Souza, Dias and Sousa Santos (2014) performed a study comparing two ray tracing selection modes in virtual reality. Two years later Dias, Pinto, Eliseu and Santos (2016) investigated two different modes of navigation and object manipulation. Their preliminary study shows that interaction via body tracking was preferred over a controller selection mode. Furthermore their results indicate that complex menus are not suited for VR as test subjects took much longer navigating through them in the non-controller setup. Their study also shows that positioning of objects via gesture interaction is much faster.

Lubos, Bruder and Steinicke (2014) performed a sophisticated study on selection errors in VR interaction depending on the view direction of the user. They found out that the selection error along the direction of view is by far the largest. They therefore suggest additional selection tolerance along the view axis.

2.2 3D Graph Layout/Drawing

One of the simplest methods for generating a graph layout usable for visualization are spring/force based layouts. They were earliest described by Eades (1984) and Fruchterman and Reingold (1991) who use equations similar to Hooke's Law to generate repulsive forces between all vertices and attractive forces between neighbouring ones. These algorithms were only designed for small graphs with up to 100 vertices. Therefore Gajer, Goodrich and Kobourov (2001) suggest an hierarchical approach to handle larger graphs. Kobourov (2012) summarises some more algorithms that work on a similar basis.

There are of course other approaches for the placement of vertices in 3D space. Reiss (1993) describes a possibility of using the Breadth- or Depth First Search trees to generate a layout. Furthermore it is possible to adapt 2D layouts to 3D space, either by mapping the 2D drawing onto the surface of a sphere as shown by Kwon, Muelder, Lee and Ma (2016) or by drawing a multiple layers of 2D drawings, which was suggested by Reiss (1993), too.

Still most graph drawing algorithms focus on a specific family of graphs beaucause it is simpler to draw a graph if some assumptions can be made about its structure.

Therefore it seems feasible to focus on the older, yet much simpler approaches as a starting point and give the user the possibility of manually adjusting the layout.

2.3 Visual Comparison of Graphs

Another aspect of this thesis is the task of visually comparing graphs. Gleicher et al. (2011) provide a good general taxonomy on different methods for visualization of differences between objects. They define three general categories of comparisons: Juxtaposition (drawing objects next to each other), Superposition (drawing objects in the same location over each other) and the explicit representation of differences.

More specific work in on visual graph comparison has been done by Andrews, Wohlfahrt and Wurzinger (2009), who have built an application called the Semantic Graph Visualizer, that allows the user to merge two graphs with common vertices drawn in 2D. They use color coding in the merged graph to indicate to which of the original graphs a vertex corresponds or if it was present in both. Brandes, Dwyer and Schreiber (2004) show a 2.5D comparison of metabolic pathways, by drawing individual graphs on layers along the third dimension. Although the use of the third dimension for comparison makes it hard to adapt it to data that already uses three dimensions, it still shows how close proximity can help with the comparison. Other research in this area is often domain specific because it is hard to find matching vertices in arbitrary graphs without additional information. This thesis will therefore focus on comparisons where corresponding vertices are known.

3 Software Description

The software written for this thesis consists of several different components, which will be described in detail in this chapter. These components are:

- The general graph visualization and vertex interaction methods (Section 3.1)
- The spring embedder responsible for automatically generating a reasonable layout for the visualization (Section 3.2)
- The UI for interaction with selected vertices and the complete graph
- Visualization of vertex attributes (Section 3.4)
- Views for comparing two graphs with common vertices (Section 3.5)

3.1 Basic interactions

The given graph is drawn with small spheres as vertices and simple straight lines as edges in between them, as can bee seen in Figure 3.1. The basic interactions that are possible with this implementation follow the same ideas as those motivated by Erra et al. (2018) and Huang et al. (2017): Vertices can be grabbed with each hand by pressing the trigger button on the back side of the controller. Further interactions were implemented but later dropped because they conflicted with the features described in Section 3.4 and were not used for the visualization of the bioecological population data, which was the focus of this thesis. Those interactions were:

- Rescaling vertices by grabbing them with both hands and pulling the hands away from each other,
- selection of vertices by touching them with the controller. Selected vertices will highlight in a different size or colour.

- Grouping of selected vertices by pushing a button on the controller. Grouped vertices are displayed as their own graph inside a semi-transparent vertex in the parent graph. An example of this can be seen in figure 3.1.

As these grouping and scaling interactions change size and colour of vertices, it is hard to combine them with the visualization of attributes described later.

What sets this software apart from the studies by Erra et al. (2018) and Huang et al. (2017) is the use of room-scale virtual reality. Instead of sitting in front of a Desk, in the case presented here, the user has an area of approximately three by four meters in which he can walk around freely. Because of this it is not necessary to use of any additional locomotion techniques, for example moving the camera forward by pressing a button on the controller. This allows for a much more natural and intuitive interaction with the virtual environment.

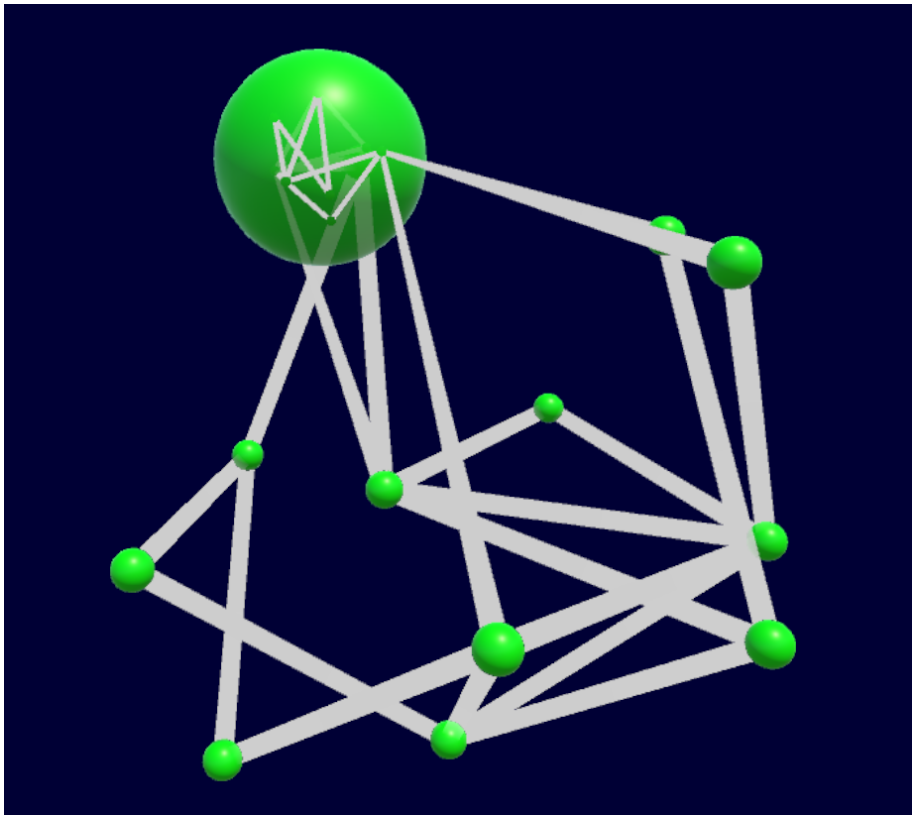

Figure 3.1. A simple graph with an embedded subgraph, The vertex that contains the subgraph becomes more transparent when the camera gets closer.

3.2 Spring embedder

In order to generate a feasible layout for visualization, a simple force-directed layout method is used, similar to the ones described by Eades (1984) and Fruchterman and Reingold (1991). In each iteration loop, it calculates repulsive forces between each pair of vertices and attractive forces for each pair of vertices that are connected by an edge. Furthermore a force is generated for each vertex that approaches a predefined bounding sphere to ensure that all vertices remain inside this sphere. This is useful to define the amount of space the graph should occupy in the virtual environment.

This relatively simple approach was chosen over more complex algorithms because in allows easy integration into the existing framework. For relatively small graphs of up to 100 vertices, these force calculations can run in real time during the engine loop which allows for continuous adjustment of the layout while the user moves around vertices as described in Section 3.1.

Vertices that are grabbed and moved around by the user have their physics simulation temporarily paused to not interfere with the drag action performed by the user, but will still influence the physics of other vertices. Therefore other vertices connected to the one moved around will follow the users movement in some way, subject to their other connections. Furthermore it is possible to pin a vertex to a certain location which will stop the vertex from getting moved around by the spring simulation while other vertices will still be affected by it. Of course, a pinned vertex can be unpinned again.

3.3 Placement in the Scene and UI

This Section is divided in two parts as there are two very different versions of UI that were developed for this project. The first iteration of the UI which consisted of radial menus attached the controller was removed after the second development cycle due to its insufficient usability and replace by the new version which only uses UI elements that are attached to fixed positions in the room.

Figure 3.2. Menus attached to the controllers. Each menu item can be used by pressing the touchpad in the corresponding sector of the circle

3.3.1 Initial Controller Based UI

Figure 3.2 shows the first iteration of the UI that uses radial menus attached to the controller which can be controlled by moving the thumb on the controllers touchpad. Until the UI was removed, only place holder graphics were used for the menu items. The menus allow for the following actions:

- Enabling and disabling a visualization of the bounding sphere around all vertices that allows the user to move and scale the whole graph.
- Enabling and disabling of a selection mode. While in selection mode, the user can select nodes by touching them.
- Grouping and ungrouping of selected vertices.
- Cycling between time steps with a forward and backward button.
- Reloading or regenerating the displayed graph

3.3.2 Second UI Version

In the second UI iteration the graph in its virtual bounding box is placed on a small table in the middle of the room. The purpose of the table is to hold UI elements for interacting with the graph. Currently it consists of only two sliders and the interface for attribute mapping described in Section 3.4. This UI can be seen in Figure 3.3

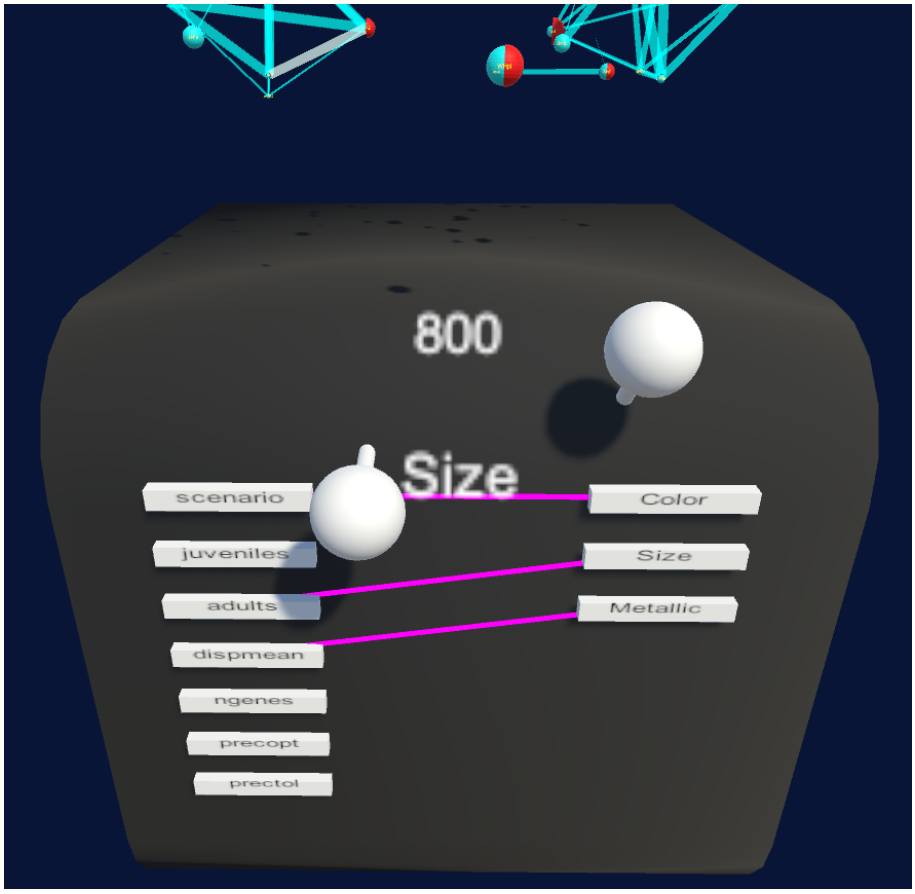

Figure 3.3. Picture of the UI. The upper slider is used for switching between graphs, the lower one is used for scaling. Below is the UI for mapped attributes

The first slider is used to cycle between different graphs that are loaded into the application to be displayed. That can for example be multiple stages of a graph that evolves over time. With the slider the user can switch between different time steps. The second slider is used to change the scale of the displayed graph to the users liking.

Both sliders can be manipulated in the same way as vertices can be moved around, but can only be moved within their sliding range.

3.4 Attribute visualization

Another feature of the application is the flexible visualisation of vertex attributes. This visualization is achieved by altering the properties of a vertex such as size, colour and shininess. Figure 3.4 shows an example of this.

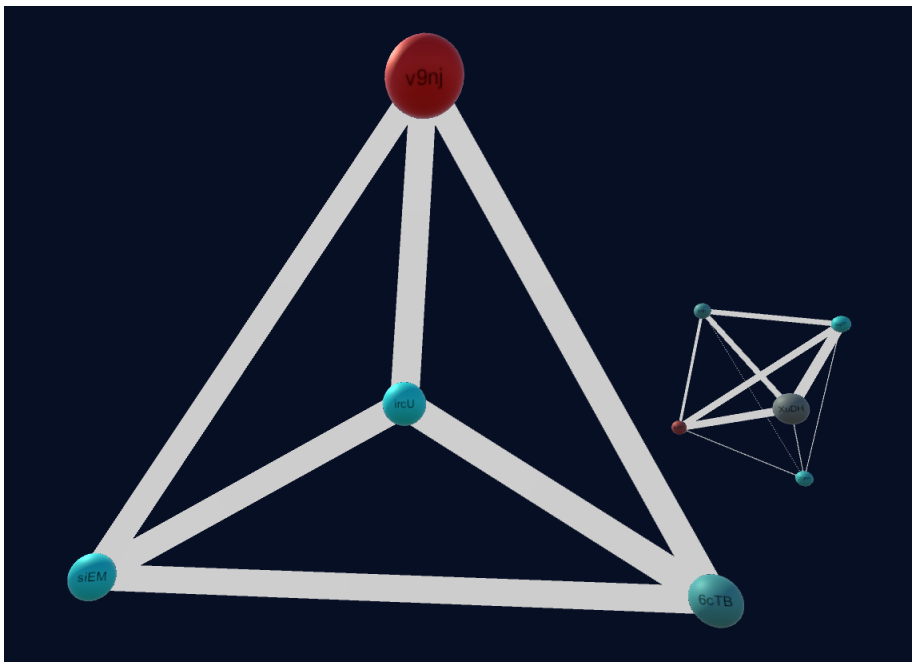

Figure 3.4. Vertices in a comparison view that have their size property mapped to an attribute

An attribute can be interpreted as a function that associates each vertex with a numerical value. There is no limit on how many attributes a single graph can have. Therefore a system has been developed to select a subset of attributes that will be visualized at the same time, limited to number of vertex properties mentioned above.

Altering the displayed subset is achieved with an UI Element that consists of two columns, as seen in Figure 3.3. While the left column contains all possible Attributes that were read from the data, the right column contains all visual properties that attributes can be mapped to, e.g. size or colour. An attribute can be connected to a visual property (always a 1 to 1 connection) by drawing a line from left to right. These connections can be changed dynamically during runtime and will become visible immediately. In accordance with the results of Lubos et al. (2014) the lines are drawn orthogonal to user standing in front of the UI and the virtual size of the

elements that is used for determining controller interaction has been increased along the view axis to accommodate for imprecise selection.

3.5 Comparison

Apart from the visualization itself, another important topic of this project is the integration of a comparison view between two graphs with common vertices.

Out of the methods of comparison described by Gleicher et al. (2011) the two main Methods *Juxtaposition* and *Superposition* are implemented. This allows for a broad comparison of these two methods in the context of visualizing graphs in virtual reality. In both cases, two scenarios consisting of multiple time steps are given. The scenarios contain common vertices that should be displayed in a comparable fashion.

3.5.1 Juxtaposition

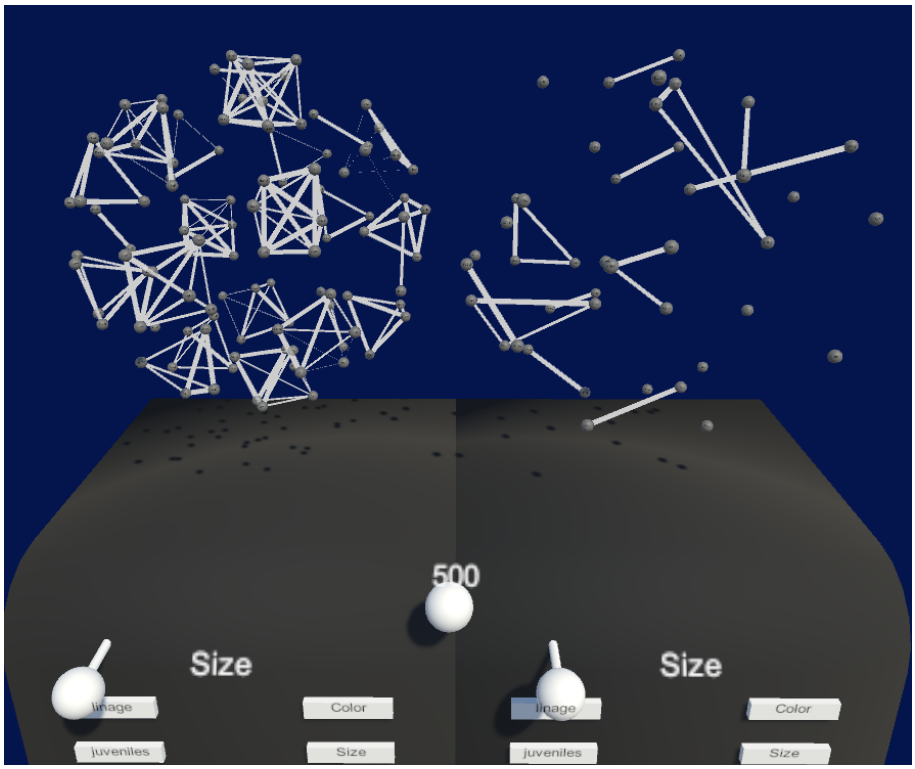

Figure 3.5. Two scenarios in juxtaposition

In this comparison view, the two scenarios are drawn side by side with common vertices drawn in the relative same location. An example of this is shown in Figure 3.5. When selecting a vertex, the corresponding vertex in the other graph is highlighted. Moving around vertices will move the corresponding vertices as well. The attribute visualization can be selected independently for each scenario but it is of course possible to select the same options for both drawings. The displayed time step is controlled by a slider for both scenarios synchronously. Furthermore, both graphs can be scaled in size.

3.5.2 Superposition

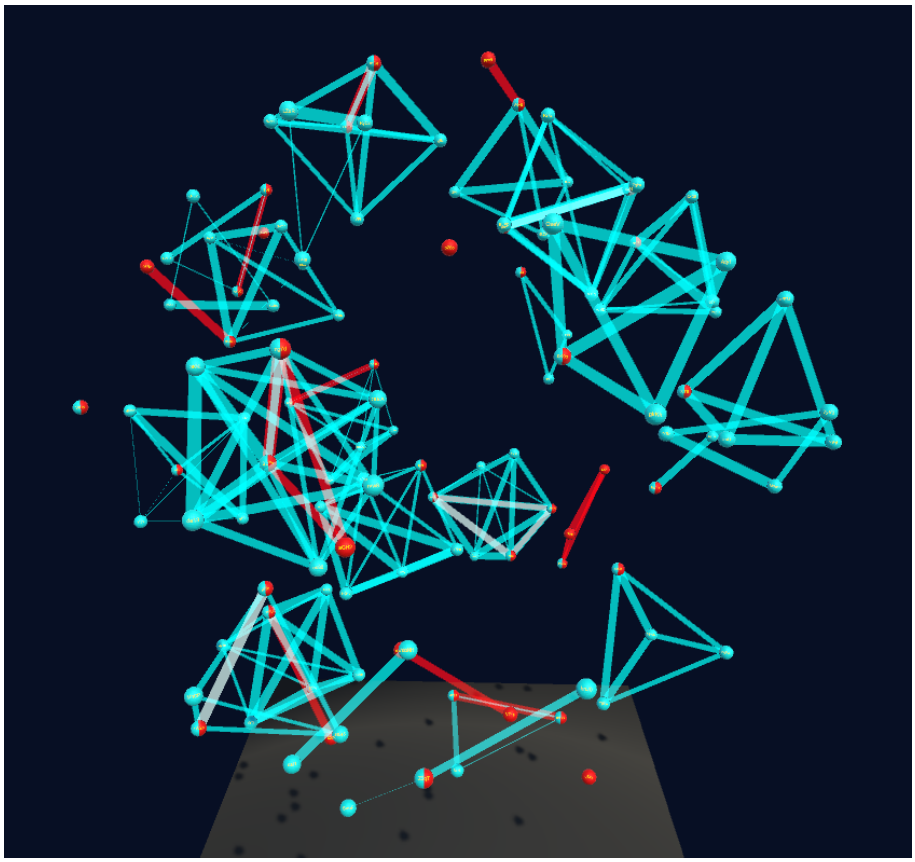

Figure 3.6. Two scenarios in superposition

In contrast to the *Juxtaposition* view, the superposition combines both scenarios into one graph. Distinction between both scenarios is done by colouring. Vertices and edges that are only present in the first scenario will be drawn in blue, those only present in the second scenario will be drawn in red. For Edges present in both

scenarios, a red and a blue line will be drawn, each with corresponding thickness. As the edges are half transparent both can be easily compared. Vertices on the other hand will be drawn of two half spheres with one coloured red and the other one coloured blue when present in both graphs. In this case the effects of attribute visualization described in Section 3.4 will apply to each corresponding half instead of the whole sphere. An example of the superposition view can be seen in 3.6

As colour is used for marking the scenario, it is reserved in this case and no longer available for attribute visualisation.

4 Application and Discussion

During the development of the visualization an iterative development process with alternating phases of implementation and evaluation has been used. For the evaluation phases, the two researchers in bioecology who have provided the data for this visualization phase were invited to test the application and give feedback on it. This feedback was gathered through informal feedback interviews and Speak-Out-Loud testing. The following sections will go into detail on each development cycle.

4.1 Initial Development and Data Specification

During the initial development phase the basic interaction possibilities, grouping and ungrouping of vertices and the spring embedder were implemented. The controller based UI was used to control these features. The main goal of this early prototype was to examine which possible datasets could benefit from the visualization. As a placeholder, a randomly generated graph with 20 vertices was used for feature demonstration. The HTC Vive[1] as the main hardware and the Unity Engine[2] as a software basis were chosen for this application.

4.1.1 Technical Foundations

The HTC Vive is one of the most popular consumer grade virtual reality devices for desktop computers and provides room scale VR. The Vive is tightly coupled with the SteamVR[3] driver by Valve which provides many useful features for application developers to interact with the device. Due to the abstraction level of SteamVR, it should be possible to run the project of this thesis on other supported devices such as the Oculus Rift with minimal effort.

[1] https://www.vive.com/eu/product/
[2] https://unity3d.com/unity
[3] https://steamcommunity.com/steamvr

As a foundation for the application the Unity Engine is used. The Unity Engine is one of the most popular game engines on the market and is not only used by many games but other projects such as data visualizations as well, for example in molecular networks, as developed by Lv et al. (2013). It is very accessible and allows for fast prototyping which makes it an ideal choice for an application like this.

4.1.2 Data Specification

The prototype was presented to the two biologists to discuss what data they could provide. After a short discussion they proposed using data from an mechanistic, eco-evolutionary simulation model. This agent-based model is stochastic, spatially explicit, grid-based, and integrates ecological (metabolic constraints, demography, dispersal, and competition), evolutionary (mutation and speciation), and environmental (geo-climatic dynamics) processes. A similar model is described by Cabral, Wiegand and Kreft (2019). Each agent has its own genome that defines its ecological behaviour, for example dispersal or environmental preferences. Agents with similar genomes belong to one species. With this model, the two bioecologists are especially interested in observing two different values:

- The strength of coexistence between species, which is determined by counting the number of grid cells in which these species coexist.
- The intra- and inter-specific combination of genomic and ecological properties.

For this manner, the model has been simulated in two different scenarios, one with and one without variation in environmental variables over time.

4.2 Integration of Data

After deciding on what data to use, a data format has been specified. In addition a method for loading the provided sample data into the existing visualization has been developed.

4.2.1 Data Format

As described in Section 4.1 the data provided for this visualization consists of two scenarios that should be compared. For each scenario, there are multiple keyframes, each describing the situation in the scenarios at that specific point of time in the simulation. While both scenarios have the same starting conditions (at time zero), they might differ severely in later time steps due to differences in environmental conditions between both scenarios. Each keyframe contains the following data:

- A vertex for each species that has alive agents in this scenario at the given point in time.
- Edges between certain species that describe the strength of their coexistence, which corresponds to the width of the edge.

This data is written into one CSV (comma separated values) formatted file in the following way: For each keyframe in each scenario, a single file is provided which contains a symmetrical adjacency matrix with edge weights that correspond to the desired width. An example of such a dataset can be seen in Table 4.1.

vertex id	0brn	1MSL	1VFt	3uGE	6cTB
0brn	0	0	0	0	30
1MSL	0	0	12	0	0
1VFt	0	12	0	0	7
3uGE	0	0	0	0	0
6cTB	30	0	7	0	0

Table 4.1. Extract from an adjacency matrix from one keyframe. The top row and the left column specify the vertex identifier while the matrix entries contain the width of edges between the vertices.

4.2.2 Integration

As a starting point, loading of a single scenario has been implemented. The loaded graph replaces the randomly generated one. As a scenario consists of multiple time steps (and therefore multiple independent graphs) a system had to be developed to display theses steps. It has been decided to display one keyframe at a time and introduce additional menu entries to the controller to allow the user to step forward

and backward through the time line. Figure 4.1 shows an example of a loaded scenario.

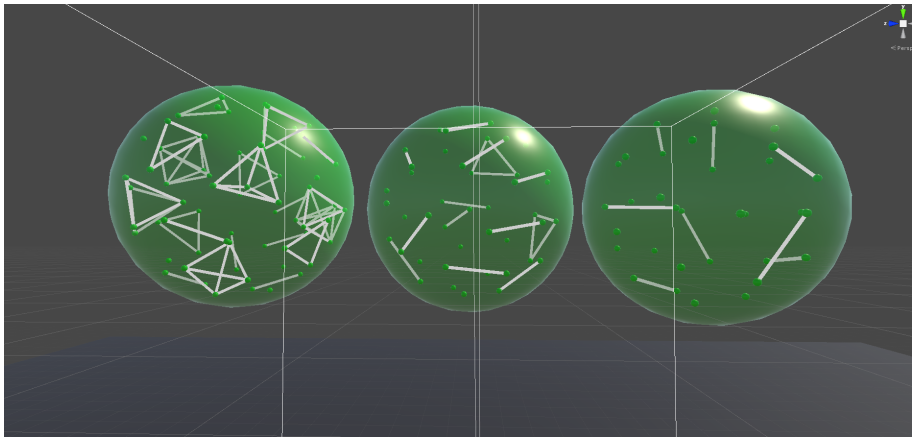

Figure 4.1. Three time steps of a loaded scenario. In the actual application these would be displayed in place one at a time by cycling through the time steps, but for the purpose of this screenshot they have been arranged side by side.

4.2.3 Evaluation

A short testing session with one of the two biologists showed many shortcomings of this initial prototype based on which the further directions of development have been defined:

- It was pointed out by the tester that while an observation of the development of a scenario over time can be interesting, the main focus of interest lies on the comparison between two scenarios at any given point in time. Therefore it was agreed that a view should be developed that allows for comparison between the two scenarios.
- While it is already a good starting point to compare the coexistences between species, all of the species have additional attributes such as the population size and its dispersal strength which would be interesting to compare visually as well.
- By observation of the test subject it became very clear that the menu based UI became too complicated and needs a replacement. It was suggested by the tester to use a slider instead. This would be especially beneficial if more time steps are added.

- The features of grouping and selection were not used by the tester and can therefore be removed in favour of a simpler UI and an easier technical implementation of the newly requested features.

4.3 Comparison and Attribute Visualization

Following the goals stated in the previous section, two separated comparison views have been implemented (see Section 3.5 for details). This was done to evaluate which of the two methods performs better. Attribute visualization was integrated into both views as an integral element of the comparison. For the Juxtaposition view this was simply done by giving each of the graphs its own attribute visualization. For the Superposition view this was harder and therefore it was only possible to visualize attributes of one scenario at a time in the early versions.

4.3.1 Data Extension

As information about attributes was not yet contained in the data, additional data was specified. Attributes contain more details about the population for each vertex, which can also change over time. These are for example the number of adults and juveniles or the dispersal strength of a population.

The attributes are read from a common file built up of many rows where each row holds the list of attributes for one tuple of vertex, keyframe and scenario. An example of such data is shown in Table 4.2.

4.3.2 General Feedback

The implementation of the described features resulted in two prototypes, one using juxtaposition and the other one using superposition. Those prototypes were tested by one of the two biologists, each one for a time of thirty minutes. Feedback was given during testing and afterwards.

In general the visualization provides a great insight into the analysed data. According to the expert it was the first time he could look at his data at such an individual

time	scenario	lineage	dispmean	juveniles	adults
200	static	0brn	0.681	235.757	4.272
200	static	1MSL	0.902	581.657	4.057
200	static	1VFt	0.667	46.090	1.363
200	static	6cTB	0.441	2959.685	21.142
200	variable	0brn	0.578	235.500	3.500
200	variable	1MSL	0.919	4626.400	29.028
200	variable	1VFt	0.634	4249.666	166.333
200	variable	3uGE	0.773	11.500	0.400
200	variable	6cTB	0.392	2117.571	23.228
500	static	1MSL	0.853	608.400	5.342
500	static	1VFt	0.640	51.416	1.875
500	static	6cTB	0.400	5790.628	41.142
500	variable	1MSL	0.931	2901.371	23.028
500	variable	6cTB	0.374	18148.600	378.400
800	static	1MSL	0.853	687.371	5.571
800	static	1VFt	0.614	80.105	2.315
800	static	6cTB	0.381	5677.828	41.085
800	variable	1MSL	0.937	214.942	1.657
800	variable	6cTB	0.370	17430.771	251.571

Table 4.2. A list of species attributes. The first two columns specify the keyframe while the third column contains the vertex id. Further columns contain one attribute each.

level. Furthermore the visualization provides a very good overview over how attributes change over time. This is possible because the selection of time steps is very intuitive: After grabbing the slider for cycling between time steps the user can focus his attention towards vertices he wants to observe. As long as he keeps holding the handle, he can still adjust the time.

What could be improved is the positioning of the sliders. At the moment they are placed on one side of the table which limits the position where a user can stand at while he wants to interact with the sliders. Therefore it would be more suitable to place the controls on a panel that is positioned relative to the user so that the controls stay in range all the time. Furthermore a feature to rotate the graph in all directions would help in accessing vertices that are at the top or back of the sphere without the need for walking around too much.

Another feature the tester requested was the ability to display detailed information about highlighted vertices. This could be achieved by temporarily displaying the

numerical values of attributes as floating numbers around a vertex as long as the user touches it with the controller. In doing so, one has to consider the current limitations of displaying text in virtual reality due to the limited screen resolution, which already makes it hard to read the labels of vertices.

4.3.3 Attributes

The attribute visualization works well both when looking at individual vertices and edges and when looking at an overview of the entire graph. One can easily spot vertices with extreme values with a glimpse at the graph but also overview the change in attributes of single vertices over time. One small problem is the scaling of data for the mapping. Currently the input values are linearly mapped to the range from 0 to 1. In cases where most input values are very small with some very extreme exceptions, most vertices are not distinguishable. While this could be solved by applying a logarithmic scale, this would break the visualization of other attributes that require a linear scale. Therefore the best scale would ideally be contained in the data.

While the visualizations of attributes works well in general, the UI for controlling which attributes should be displayed turns out to be more counter-intuitive and tedious than expected and should be reworked. A possible solution for this would be to use some sort of rotatable selectors, one for each visual property.

4.3.4 Automatic Layout

As detailed in Section 3.2 an automatic layout of the graph is achieved by using a simple spring simulation. When this simulation is run in real time, which is feasible for up to 100 vertices, other vertices will dynamically adapt to rearrangements by the user. This is especially helpful as it allows entire groups of strongly connected vertices without the need for an dedicated group moving ability and therefore results in a simpler usability.

Due to the time complexity of the spring simulation in $O(n^2)$, it is not feasible to run the computations every frame for a larger amount of vertices. Therefore in the case of larger graphs the simulation is only run during startup of the application to create an initial layout and is disabled afterwards. A possible improvement for

medium sized graphs would be to run the simulation in a background thread and only periodically update the visualization. This would of course be not as smooth but still provide some interactivity.

One could also consider using different and more complex layout algorithms with better time complexity that can handle larger amounts of vertices with reasonable amounts of computation time.

Another solution would be to encode positional data in the input files and get rid of automatic layout completely or at least partially. Vertex positions could even be used as an additional option for attribute visualization so more attributes can be compared simultaneously.

4.3.5 Comparison View

As mentioned in Section 3.5, two different views for the comparison of two scenarios have been implemented: Juxtaposition and Superposition. While it works fine for comparing differences between populations within a scenario, for the main task of making it easily possible to compare between two scenarios, the juxtaposition view has some problems.

- The possibility for scaling each graph independently turns out to be a disadvantage because the lack of a synchronized size makes comparing size mapped attributes very hard.
- The distance between the two graphs is a bit to large (especially when scaled down) which does not always allow for a decent overview, especially as it is not always clear which vertex belongs to which graph. These issues could be fixed by scaling the distance between the graphs according to their scale and drawing a bounding box around each one.
- It is tedious to assign the attributes separately for each graph as most of the time, one wants to compare the same attributes on both graphs
- The juxtaposition does not cope well with the current layout algorithm (Described in Section 3.2), which is only designed to layout on graph at a time. That means that only one graph is layouted and the positions of vertices in the other graph are synced to that. This produces a rather bad layout with vertices that only exist in the second graph.

This issue could potentially be resolved by generation of a virtual union graph to use with the layout algorithm and then apply the computed position to both visualizations.

While all these issues can be at least partially resolved, it turns out that it is much simpler to switch to the superposition view instead, which does not have these four drawbacks. In general the superposition view allows for much better comparison because the user can focus on just one point in the scene instead of two. For example one can easily spot how big the difference in thickness of an Edge is between the two scenarios as can be seen in Figure 4.2. Therefore it is much more clear where

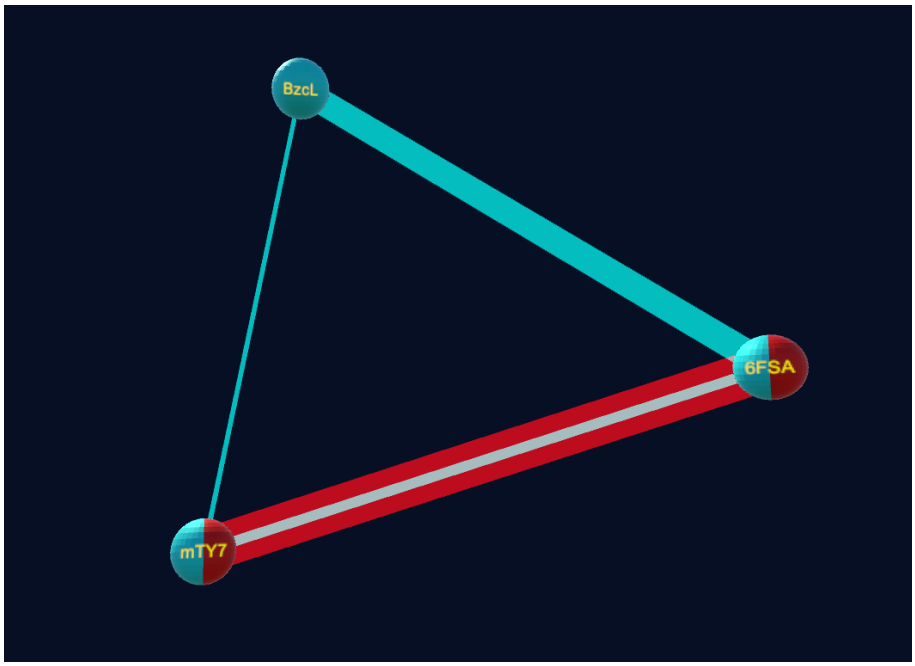

Figure 4.2. Example of an edge that has different width in both scenarios.

the coexistences of species are different and where they are similar.

One big drawback of the superposition view during the testing session was the lack of a possibility to visualize attributes of both graphs at the same time, as a vertex could only have one size and one colour. This was later resolved by introducing the concept of using half spheres to visualize the attributes, with each half sphere corresponding to one scenario. This separation of vertices into two halves to visualize attributes of both scenarios simultaneously proves to work well for comparison, an example of this can be seen in Figure 4.3

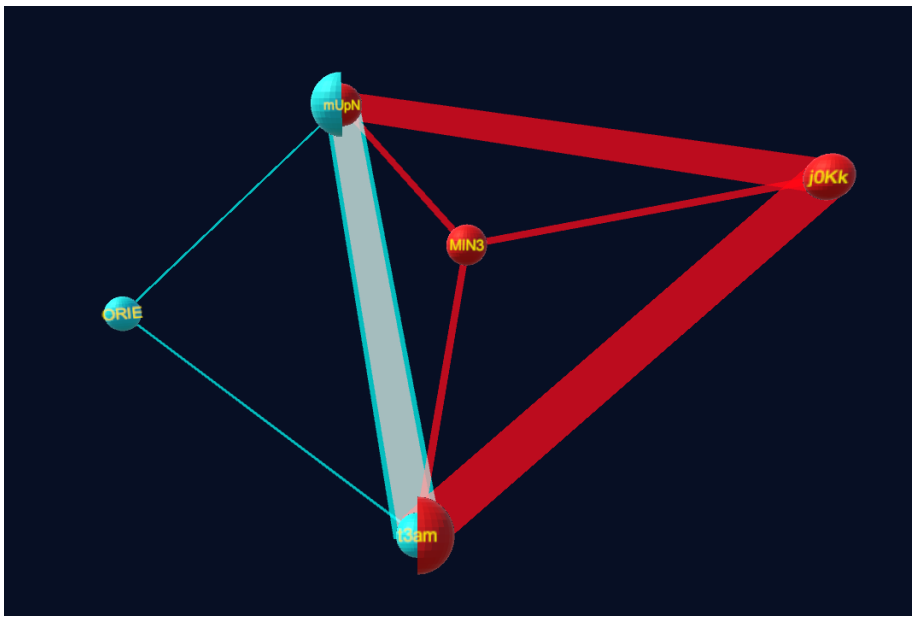

Figure 4.3. Example of vertices that have different attributes for each scenario.

An additional feature that would be very useful for this view according to the tester is the possibility of temporarily disabling the one of the two graphs to have a closer look at the other one without visual interference.

5 Conclusion

5.1 Findings

Virtual reality is becoming more and more common in data visualization. While there are still hurdles to overcome and the hardware is not perfect yet, it allows us to create more immersive visualizations that help in the analysis of data. In this thesis it has been explored how graphs can be displayed in virtual reality and what possibilities exist to visualize certain additional attributes. Furthermore possibilities of creating an intuitive scenario for comparing two graphs with common vertices have been developed.

The visualization has been tested with data from biological population development simulations to find out which aspects work well and which would require improvements if this software was to be used in production. Nevertheless the great potential that this visualization has for the analysis of the predescribed data has been shown. According to the bioecologists they got a quick insight into the coexistence and survival time of the simulated species. Especially the differences between the two scenarios could be spotted very quickly.

5.2 Future Work

While Chapter 4 has already detailed a lot of aspects in which the visualization presented in this thesis could be improved, there are some further aspects that could be looked at in more detail.

One of these aspects is providing a more interactive the layout of the graph. For example, selected vertices could be moved to the front while others that are not in the neighbourhood are moved further back. Completely different layout techniques could be used as well such as positioning vertices in a sphere around the user as done by Kwon et al. (2016). As part of exploring different layouts, grouping and

the separation of sub-graphs could be reintroduced as well. This might be especially useful as the visualization clearly showed that most of the strong coexistence relations are grouped into small complete sub-graphs meaning that throughout the simulation, agents of the same group of species tend to coexist. Therefore it should be considered if the visualization via a simple graph is really the best solution for this kind of data. Shneiderman and Dunne (2012) present the idea to use different geometric shapes ("motifs") to represent certain complete sub-graphs. A similar technique using for example platonic solids could possibly be applied here. Finding an improved graph structure will become even more important to retain a good overview if much larger data sets should be used.

Another aspect that could be worked on is the comparison. Currently it aims at showing both datasets in an optimal way to make it as easy as possible for the user to spot differences. Instead one could create a visualization that explicitly shows the differences instead of the actual data.

Finally the findings of this thesis are mostly based on qualitative observations that give good insights but no definitive results. Therefore it would be necessary to conduct user studies on certain aspects such as the UI or different layouts and comparisons to get quantifiable results.

Bibliography

Andrews, K., Wohlfahrt, M. & Wurzinger, G. (2009). Visual graph comparison. In *13th International Conference Information Visualisation* (pp. 62–67).

Brandes, U., Dwyer, T. & Schreiber, F. (2004). Visual understanding of metabolic pathways across organisms using layout in two and a half dimensions. *Journal of Integrative Bioinformatics*, *1*.

Cabral, J. S., Valente, L. & Hartig, F. (2016). Mechanistic models in macroecology and biogeography: State-of-art and prospects. *Ecography*, *10*.

Cabral, J. S., Wiegand, K. & Kreft, H. (2019). Interactions between ecological, evolutionary, and environmental processes unveil complex dynamics of island biodiversity. *Journal of Biogeography*.

Dias, P., Pinto, J., Eliseu, S. & Santos, B. S. (2016). Gesture interactions for virtual immersive environments: Navigation, selection and manipulation. In S. Lackey & R. Shumaker (Eds.), *Virtual, Augmented and Mixed Reality* (pp. 211–221). Lecture Notes in Computer Science. Springer International Publishing.

Eades, P. (1984). A heuristic for graph drawing. *Congressus numerantium*, *42*, 149–160.

Erra, U., Malandrino, D. & Pepe, L. (2018). Virtual reality interfaces for interacting with three-dimensional graphs. *International Journal of Human–Computer Interaction*, *35*(1), 75–88.

Fruchterman, T. M. J. & Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, *21*(11), 1129–1164.

Gajer, P., Goodrich, M. T. & Kobourov, S. G. (2001). A multi-dimensional approach to force-directed layouts of large graphs. In J. Marks (Ed.), *Graph drawing* (pp. 211–221). Springer. Berlin, Heidelberg.

Gleicher, M., Albers Szafir, D., Walker, R., Jusufi, I., D. Hansen, C. & Roberts, J. (2011). Visual comparison for information visualization. *Information Visualization*, *10*, 289–309.

Huang, Y.-J., Fujiwara, T., Lin, Y.-X., Lin, W.-C. & Ma, K.-L. (2017). A gesture system for graph visualization in virtual reality environments. In *IEEE Pacific Visualization Symposium (PacificVis)* (pp. 41–45).

Kobourov, S. G. (2012). Spring Embedders and Force Directed Graph Drawing Algorithms. *arXiv:1201.3011 [cs]*. arXiv: 1201.3011.

Kwon, O.-H., Muelder, C., Lee, K. & Ma, K.-L. (2016). A study of layout, rendering, and interaction methods for immersive graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, *22*(7), 1802–1815.

Lubos, P., Bruder, G. & Steinicke, F. (2014). Analysis of direct selection in head-mounted display environments. In *IEEE Symposium on 3D User Interfaces (3DUI)* (pp. 11–18).

Lv, Z., Tek, A., Da Silva, F., Empereur-mot, C., Chavent, M. & Baaden, M. (2013). Game on, science - how video game technology may help biologists tackle visualization challenges. *PloS One*, *8*(3), 1–13.

Millais, P., Jones, S. L. & Kelly, R. (2018). Exploring data in virtual reality: Comparisons with 2D data visualizations. In *Conference on Human Factors in Computing Systems* (LBW007:1–LBW007:6). CHI EA '18.

Reiss, S. P. (1993). A framework for abstract 3D visualization. In *IEEE Symposium on Visual Languages* (pp. 108–115).

Shneiderman, B. & Dunne, C. (2012). Interactive network exploration to derive insights: Filtering, clustering, grouping, and simplification. In *International Symposium on Graph Drawing* (pp. 2–18). Springer.

Souza, D., Dias, P. & Sousa Santos, B. (2014). Choosing a selection technique for a virtual environment. In R. Shumaker & S. Lackey (Eds.), *Virtual, augmented and mixed reality. Designing and developing virtual and augmented environments* (pp. 215–225). Lecture Notes in Computer Science. Springer International Publishing.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Würzburg, 10.05.2019

Michael Kreuzer