VRige: Exploring Social Network Interactions In Immersive Virtual Environments

Adam Drogemuller*, Andrew Cunningham*, James Walsh*, William Ross†, Bruce H. Thomas*

*Wearable Computer Lab, University of South Australia, Adelaide, Australia
†Department of Foreign Affairs and Trade, Canberra, Australia

Email: adam.drogemuller@mymail.unisa.edu.au, {andrew.cunningham, james.walsh, bruce.thomas}@unisa.edu.au

Abstract—Analysis of social networks is an active research area for a large number of fields. Whilst advances in machine learning provided analysts with a wider range of automated tools, there still remains a significant amount of analysis that is performed through the visualisation of the networks. Visualization allows the user to identify clusters, patterns, and anomalies within a dataset. In this paper we introduce VRige, a virtual reality system for visualization, exploration, and tagging of social networks design in Unity 3D. Our new system allows for the exploration both the connections between entities and the entity attributes. In this paper we provide an overview of VRige, the visualization design, interactions, and system design. We also describe a number optimizations we made in order to support a system such as this in Unity 3D.

Index Terms—virtual reality, visualization, social computing

I. INTRODUCTION

Network visualisation is a key task in law enforcement and regulatory agencies. Traditionally, analysts build network charts by hand using tools such as Analyst Notebook [1] to collate information, identify relationships and activities, and communicate the importance of particular individuals in relation to an investigation. As the volume of data increases, this bottom up approach becomes less viable, and it has become more common for organisations to provide network visualisation tools as entry (and aggregation) points to data stores, enabling analysts to visualise relationships in data already held, and (in some cases) to contribute new information as the investigation develops. These tools typically present a two-dimensional view and rely on a flat-screen presentation. Whilst this is sufficient for investigator-led inquiry where the analyst “walks-out” from known starting points to grow a picture of the investigation, they are inherently limited in their ability to support a top-down examination of complex or highly connected data.

With the commodification and recent advancements in Virtual Reality (VR), there has been a renewed interest in exploring data visualisation through the immersion ability of VR technology [2] to afford presence. Consequently, recent research has begun re-examining graph visualisation in a VR context [3]–[5]. Motivated by this renewed interest in VR graph visualisation, we have been exploring VR social network graph visualisation [6] in the domain of law enforcement, addressing some of the unique requirements that the domain presents.

Through structured interviews with law enforcement workers, we have identified several of these unique requirements. The visualization need to support the user to easily identify associations/links in a network of people, and specifically outliers/singleton such as persons not connected to any other person. Attribute information has to be easily comparable between persons $a$ and $b$ specifically, in order to determine if two people have a strong connection (i.e., two persons are within the same location, and the same circle of people). Filtering of nodes and communities is an important goal for the project, in regards to hiding information and entities not of interest to the user so that the user can focus on points of interest in the network.

In this paper, we present VRige (Virtual Reality Immersive Graph Explorer), our preliminary exploration into immersive VR social network graph visualisation with a complimentary set of rich interactions. VRige employs a spring embedder layout and a virtual, direct input widget for filtering the graph based on social network tasks. We present our early observations of this system, and conclude with a discussion of future work.

II. BACKGROUND

A primary topic in graph visualisation research is layout algorithms. Eades and Tamassia [7] identify several heuristics that optimise graph layouts for readability, including avoiding edge crossings, keeping edge lengths uniform, and distributing node uniformly. Force-directed layouts, whereby simulated physical forces are used to manipulate the position of nodes within a graph, have been shown to optimize for these heuristics. A seminal example of force directed layout is the spring embedder layout, first proposed by Eades [8]. In a spring embedder layout, edges between nodes are treated as springs approximating Hooke’s law to maintain an optimal edge length while repulsion forces are applied between nodes to reduce overlap and prevent the system from collapsing. Although research has shown that the choice of layout can have a prominent effect on understanding a social network [9], at this early stage in our research our focus is on exploring immersive interaction techniques that support social networks. As such, we have implemented a spring embedder layout in our system as it readily adapts to 3D and supports the
interactions we are seeking to explore in immersive VR environments.

*Immersion* and *presence* are well-established attributes within VR research. *Immersion* is the capability of a technology to deliver an illusion of reality [10]. *Presence* is the subjective sense of being in the virtual environment and is therefore directly affected by *immersion* [10]. Ware and Mitchell [11] studied the effects of various immersive cues in perceiving 3D graph projections and found a significant increase in the ability to trace graph paths when using these cues compared to 2D screens.

High-resolution tiled displays have been shown to enhance perception and navigation for visual tasks by Ball and North [12]. Their study showed that larger display area and resolution positively affect users capacity to find and compare visual targets. Shupp et al. [13] investigated viewport size and curvature of large high-resolution displays. They determined that curved tiled displays amplified user performance for route and target search tasks in the context of map visualisation. Virtual environments have been shown to be effective in many scientific applications such as archaeology [14], brain tumour analysis [16], geographic information systems [17], geosciences [18], [19] or physics [20]. Chandler et al. discussed the potential impact of immersive environments on the engagement of people with data and with each other in data analysis and decision making, defining a research thrust they call *Immersive Analytics* [2].

The recent availability of advanced VR technologies means immersive graph visualisation can now be explored again. Following from previous works, we utilise previous research [21] in exploring how it can be used to support the exploration of social graphs in VR.

III. Social Network Visualisation

Our visualization system, VRige (Figure 1), produces 3D layouts of undirected social network graphs in a VR immersive environment. These social networks are comprised of people, or “entities”, represented as spherical *entity nodes*, abstract relationships, represented as line *edges*, and social clusters, or “circles”, represented as *cluster nodes*. Nodes have a text label indicating their ID. Cluster nodes are 50% larger than entity nodes and are assigned a randomised colour generated with a constant saturation and luminance with a random hue value. Entity nodes inherit the color of their cluster node, or, if no clustering has been performed, are populated with a default blue colour. The graph layout is performed by randomly placing the nodes within a cubic volume and then running a 3D spring embedder layout with the unique rest condition of all edges being at a length $\leq 1$.

The system is built in the Unity 3D engine and uses the HTC Vive headset and controllers. A blue cone, depicted in Figure 2b, is attached pointing away from the controller to aid with precise selection and interaction, as described below.

To alleviate simulator sickness when traversing the graph, we inserted four grey panels throughout the scene on the boundaries of the render area, as depicted in the background of Figure 1. By using these passive markers in the background, we provide a visual grounding to the users. This provides a frame of reference to provide the sensation that the graph is moving within the space, rather than the user. Whilst textured backgrounds or more elaborate skyboxes could have been used, using a simple background ensures the graph maintains the user’s focus.

IV. Interaction Techniques

In order to explore the graph, a number of interaction techniques are supported to allow users to focus, filter, and show
details-on-demand; these techniques support Schneiderman’s Visual Information-Seeking Mantra, “Overview first, zoom and filter, then details-on-demand” [22]. Aside from the user being able to simply point/grab/drag clusters around within 3D space using the trigger, the ability to rapidly filter the dataset based on specific criteria are supported through a six-sided Filter Cube.

A. Navigation and Interacting With Nodes

Users are free to walk around the graph in the VR environment. The graph can be independently rotated by holding a grip button on the side of the right Vive controller. While holding the grip button, the relative rotations of the controller directly maps to the rotation of the graph. This provides users a method to rotate the graph independent of their viewpoint, thereby alleviating equatorial bias (where nodes and their relationships are more prominent along the horizontal of the graph) that may be imparted by a 3D spring embedder layout.

Holding down the touchpad emits a laser pointer (Figure 2b) which can be used to select and highlight nodes within the graph. Users can select a node by pointing the the laser pointer at the target node and pressing the controller trigger, which will either expand a cluster node or highlight a entity node, depending on what type of node was selected. Using this interaction on an already-highlighted node toggles the node’s selected state.

When a particular node is highlighted and the user comes in close proximity to that node, the node’s associated attributes are presented in a 3D pop-up window (Figure 2a). Within the window, the attributes are presented as labeled values within a scrolling list, and the user can scan through attributes by colliding and swiping the controller up and down the list. The attributes window can be closed by hitting the “X” button at the top right of the window.

Entity and cluster nodes can be directly interacted with by intersecting the blue selection cone of the controller and holding the trigger. Nodes can then be grabbed and moved, allowing users to construct and organize nodes into custom visual groups and change the overall layout to fit their requirements of the graph.

B. Filter Cube

The Filter Cube (Figure 3) is a virtual object that can be used to filter and highlight nodes in the graph with specific attributes. The cube is attached to a Vive tracker, so picking
up the tracker in the real world moves the cube in the virtual environment. Thus, the user must physically rotate the tracker or reposition their head relative to the hand in order to perceive all sides of the cube.

Emulating a Fidget Cube, users interact with the buttons/sliders on each side of the cube to select attributes and change the filter. Each side of the cube corresponds to a different attribute category, as denoted in Table I. The user can press the power button on each side of the cube to toggle the filter for that side. However, given the number of nodes present in the datasets, changes to the filter cube are not dynamic, rather, the user can press the query button to apply the current filter cube settings to the graph. If all power buttons are off and the query button is pressed the graph will revert back to its original, unfiltered, state.

The filter cube allows the user to only have to focus on the user interface when required, allowing persistent focus on the graph instead of the user interface. Whilst the use of the cube physically limits the number of filters available to the user, it encourages users to playfully explore the dataset.

C. Saving and Exporting

In acknowledging the fact that the system is not used in isolation, we support to ability for users to select, save, and export nodes of interest. Using a physical storage box metaphor, users can drag nodes of interest into a virtual storage box (Figure 4), allowing for later retrieval and highlighting, or exporting of those nodes out of the program via CSV. In using this metaphor, the user can have multiple boxes, each storing/representing a different selection of nodes of interest. The boxes appear on a virtual table, that is actually a virtual representation of a physical table, that we track using a supplementary Vive tracker.

![Fig. 4: The storage box, used to save a collection of nodes by physically placing them within the box.](image)

Fig. 4: The storage box, used to save a collection of nodes by physically placing them within the box.

---

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>Continuous</td>
<td>20.00..90</td>
</tr>
<tr>
<td>Location</td>
<td>Categorical</td>
<td>Adelaide, Melbourne, ...</td>
</tr>
<tr>
<td>Education</td>
<td>Categorical</td>
<td>High school, Tertiary, ...</td>
</tr>
<tr>
<td>Occupation</td>
<td>Categorical</td>
<td>Designer, Developer, ...</td>
</tr>
<tr>
<td>Family Name</td>
<td>Categorical</td>
<td>Brandon, Lee, ...</td>
</tr>
</tbody>
</table>

---

![Fig. 5: Basic architecture diagram showing the relationship between the various system components.](image)

Fig. 5: Basic architecture diagram showing the relationship between the various system components.

V. SYSTEM OVERVIEW

VRige employs a Model-View architecture, separating the underlying data from its representation, as depicted in Figure 5. As previously described individually, the main components of VRige consists of:

- Nodes component: The coloured spheres representing entities.
- Cluster nodes: A group of node components.
- Edges component: The links between nodes.
- Edge renderers: The graphics renderer responsible for edge display.
- Force directed edges: Responsible for the layout of the nodes.
- Clustering component: Responsible for the node clustering based on user-defined values.

When opening a dataset, initialising the data structures creates a number of node objects associated with one graph object which contains all the data node objects for the data model. In generating the visualisation, nodes in the model are clustered, and created as part of the view with a cluster renderer that shows the nodes at 150% the normal node size. The remaining nodes are then shown in the view with Visual Nodes created to render them. Force Directed Edges (FDEs) are then instantiated and produce a force-based layout between nodes, with the edges deleted when the layout physics complete. Given the FDEs are destroyed, a separate, persistent Visual Edge object is created to render the edge both during the initial layout phase, and after the FDEs have been destroyed.

A. Dataset

VRige accepts two dataset files; an entities and edges data file, and an attributes data file. The entities and edges file may be in a “circular” format, where the node clusters have been pre-calculated into “social circles”, or a more general unclustered format. For VRige application to recognise circular formats, the data file must contain a preamble containing a
list of cluster identifiers and their associated entity nodes. Otherwise, the program will recognise the dataset as a general undirected network. In the case of a general network, VRige will run a K-Means Clustering to synthetically define these circles, with:

$$K = \frac{|\text{entities}|}{\mu}$$

Where $\mu$ is an arbitrary divisor. From our experience, for graphs in the order of 1000 to 9000 nodes, we have settled on $\mu$ value of 100.

VI. Optimisations

Whilst Unity was originally developed as a game engine, it is seeing increasing use in a plethora of domains, including 3D visualisation work. However, there is a number of issues Unity presents when working outside its core domain.

Despite running on a high end graphics card (NVidia GTX970), there was a significant impact on rendering performance for even a reasonably-sized social-network dataset. We found that rendering a considerable amount of objects (such as 1000 nodes) individually in a scene caused considerable performance issues such as the application rendering below 30 FPS, which is suboptimal for immersion within a VR display. We found that if the nodes in the application were batched together into singular objects based on clustering, performance increased substantially to the optimal 90 Hz. For example, if there were 10000 nodes in the environment, rather than instantiating 1000 individual objects, we can batch node objects together into ten objects such that one object contains 100 nodes. Users can then expand a cluster so that they can interact with each node. The clustering of objects is determined using K-Means Clustering if the data is not pre-clustered. Random colours are also assigned to each cluster in the environment to separate them visually. The colours for each cluster are also assigned for each edge to improve readability and separation of clusters visually in the program.

Edges are rendered on top of the visualization on a single component using OpenGL instead of within the Unity Engine, which was found to increase performance significantly.

VII. Conclusion and Future Work

In this paper we described VRige, a virtual reality graph visualiser designed for social networks. We presented a number of novel interaction techniques, allowing users to view, filter, and export nodes of interest in large scale social network graphs, implemented in Unity 3D. We described a number of optimizations we made in order to allow Unity to support the requirements for a system as this. Future work will include the further development of interaction (such as hand-based techniques [23]), and user evaluation of the system.

Acknowledgment

This work has been supported by the Data to Decisions Cooperative Research Centre whose activities are funded by the Australian Commonwealth Governments Cooperative Research Centres Programme.

References
