Summer 2017

GENE EXPRESSION PROSPECTIVE SIMULATION AND ANALYSIS USING DATA MINING AND IMMERSIVE VIRTUAL REALITY VISUALIZATION

Joshua Cotes
Eastern Washington University

GENE EXPRESSION PROSPECTIVE SIMULATION AND ANALYSIS USING DATA MINING AND IMMERSIVE VIRTUAL REALITY VISUALIZATION

A Thesis

Presented To

Eastern Washington University

Cheney, WA

In Partial Fulfillment of the Requirements

for the Degree

Master of Science in Computer Science

By

Joshua M. Cotes

Summer 2017

THESIS OF Joshua M. Cotes APPROVED BY

_____ Date_____
Dan Tappan, GRADUATE STUDY COMMITTEE

_____ Date_____
Stu Steiner, GRADUATE STUDY COMMITTEE

_____ Date_____
Uri Rogers, GRADUATE STUDY COMMITTEE

Abstract

GENE EXPRESSION PROSPECTIVE SIMULATION AND ANALYSIS USING DATA MINING AND IMMERSIVE VIRTUAL REALITY VISUALIZATION

by

Joshua M. Cotes

Summer 2017

Biological exploration on genetic expression and protein synthesis in living organisms is used to discover causal and interactive relationships in biological processes. Current GeneChip microarray technology provides a platform to analyze up to 500,000 molecular reactions on a single chip, providing thousands of genetic and protein expression results per test. Using visualization tools and priori knowledge of genetic and protein interactions, visual networks are used to model and analyze the results.

The virtual reality environment designed and implemented for this project provides visualization and data modeling tools commonly used in genetic expression data analysis. The software processes normalized genetic profile data from microarray testing results and association information from protein-to-protein databases. The data is modeled using a network of nodes to represent data points and edges to show relationships. This information is visualized in virtual reality and modeled using force directed networking algorithms in a fully explorable environment.

Chapter

List of Figures

Acronyms

3D	Three Dimensional
APK	Android Package Kit
BioGRID	Biological General Repository for Interaction Datasets
cDNA	Complimentary DNA
DNA	Deoxyribonucleic Acid
GEO	Gene Expression Omnibus
GUI	Graphical User Interface
HTML	Hypertext Markup Language
mRNA	Messenger Ribonucleic Acid
NCBI	National Institute for Biotechnology Information
REST	Representational State Transfer
RNA	Ribonucleic Acid
UI	User Interface
URL	Uniform Resource Locater
VR	Virtual Reality

1 Introduction

1.1 Problem Statement

Computational methods for identifying patterns and making predictions on data using machine learning algorithms, or data mining, can give insight into large quantities of data that would not be practiced without computer processing. These algorithms use mathematical tools such as statistics to assess patterns in data and provide a human-understandable representation of those patterns. When datasets become large, visual interpretation of data can become overcrowded with information and be difficult to understand, especially using two dimensional representations. One specific challenge is extracting usable data from biological genetic expression testing and identifying biologically relevant patterns from that data.

Biological genetic expression profiling is used to determine the relative quantity of specific proteins present in a biological organism during gene expression conditions. Results are obtained using microarray expression profiling, a high throughput method of determining protein quantities in a biological sample [2]. The data obtained from microarray processing is the result of specified bait-protein markers binding with proteins in the test solution, structured as a list of proteins with quantitative measures of bait-protein binding for each protein. A challenge with interpreting microarray profiling is the quantity and dimensionality of the data, which can have hundreds of results for a single expression test. Unsupervised data mining algorithms such as k-means clustering can organize and provide useful information about the data by grouping similar subsets of proteins from the dataset.

Protein and gene association data can provide additional information about which proteins in a dataset are required together during gene expression activity. Association data provides information about the data by indicating if proteins work together in a biological process [3]. Adding association information

to data mining visualizations provides additional knowledge about interactions that would not be present with data mining alone.

1.2 Goals

The objective of this project is to build a modeling, visualization, and simulation system used to analyze biological protein expression profiling data and association information using Virtual Reality (VR). The VR platform uses three dimensional space for viewing images, providing additional space to render viewable datapoints. The ability to move around in three dimensions of visualized data in VR provides additional data viewing methods not possible in two-dimensional representations, allowing access to all datapoints independently.

K-means clustering can be used to determine the grouping patterns in the data for analysis [4]. This method is unsupervised and needs no user set parameters during clustering to operate, allowing the user to view groupings and gain information almost immediately. Cluster visualization in three dimensions provides a unique way to understand the data by separating clusters into different three-dimensional spaces. Where two-dimensional representations may become unintelligible when datasets are large, the virtual reality representation allows the user movement to any location bringing hidden clusters into view.

Data records for expression profiling typically have multiple test results, or attributes, indicating test values related to that record. Interpreting these values quickly in VR is possible by changing the color of all datapoint objects in the scene to represent one of those attributes. This is done by the user setting an attribute as the focus and updating all datapoint objects of the change. Combining color attribute focus on datapoints with association information can provide additional information about the expression values of a datapoint related to associations with other proteins.

The final goal of the project is to provide an analysis tool that provides information about the data from different perspectives. A combination of k-

means, association data, focus attribute setting, and datapoint creation in a three-dimensional environment provides information about data not available using any single method. The features of the program can be used independently or in combination, allowing for many different analysis techniques.

2 Background

2.1 Gene Expression

Deoxyribonucleic acid (DNA) encodes the genetic instructions in all living organisms and some viruses. DNA consists of two complimentary strands of nucleotide polymers that are paired together in a double helix. The nucleotide polymers are made up of the monomer chemical bases: adenine, cytosine, guanine, and thymine (A, T, G, and C) [5]. The ordering of the base sequence determines the information needed to synthesize parts of the organism. Pairing of the polymers is specific, forming base pairs of adenine with thymine and guanine with cytosine. The sequence of bases acts as a template for duplication of the DNA strand in the DNA replication process and a template for Ribonucleic acid (RNA) synthesis in gene expression [5]. During cell replication, strands of DNA are separated and transcribed, resulting in an identical copy of the original strands passed to the child cell.

Gene expression is the process of decoding the base sequence of DNA and translating the result into a functional product. The resulting products of translation are proteins and functional RNA, determined by the gene nucleotide sequence. Ribonucleotides make up RNA, similar to those found in DNA. The ribonucleotides found in RNA include: adenine, guanine, cytosine, and uracil [5]. Ribonucleotides pair with DNA nucleotides to form a complimentary RNA strand of uracil with adenine and guanine with cytosine. Unlike DNA, RNA forms a single strand and can be used during the translation process as messenger RNA (mRNA). During transcription, the DNA molecule base pairs are locally separated and transcribed into RNA. The process is catalyzed by the enzyme RNA polymerase, which is responsible for separating the DNA strands and adding ribonucleotides complimentary to the DNA nucleotides on the new RNA strand [5]. If the gene encodes a protein, the resulting RNA is mRNA and is used by cell ribosomes, a protein structure, to synthesize a chain of amino

acids. The ribosome processes the mRNA by moving along the mRNA strand and adding amino acids to a new chain. Each set of three ribonucleotides in the mRNA strands encodes a specific amino acid to add to the chain. The amino acid chain then folds into a protein and is used by the cell to perform a function [5].

2.2 Microarray Data

The DNA microarray has become a tool primarily used to discover and identify genetic mutations as well as the expression level of proteins in biological systems. Microarray testing is a high throughput testing method for isolation and identification of the presence of genetic sequences and protein encoding RNA strands [2]. The microarray is typically a glass slide to which DNA nucleotide subsequences are affixed. The nucleotide subsequences are arranged on the slide as binding groups in a two-dimension array, which acts as a substrate for the testing material to bind in a process called hybridization. Hybridization occurs when complimentary strands of RNA and DNA make contact and bind together or to a bait antibody [2]. Microarray slides are prepared by creating thousands of microscopic wells using a robot or the printing process of photolithography [2]. Millions of copies of the DNA strands are produced using polymerase chain reaction (PCR) and affixed in matching groups to the wells. The position of the groups in the array are used to identify the hybridized strands.

Microarray testing used to identify the presence of genetic mutations is performed by growing cells in a laboratory and hybridizing the cells RNA to the microarray slide. Typically, two sets of cell lines are grown for test, one used is a reference and the other is the test case from an organism. The genetic material is extracted from both cell lines then transcribed into a complimentary structure called complimentary DNA (cDNA) [2]. Transcription is necessary because the microarray substrate material is a template of the DNA to be identified and will

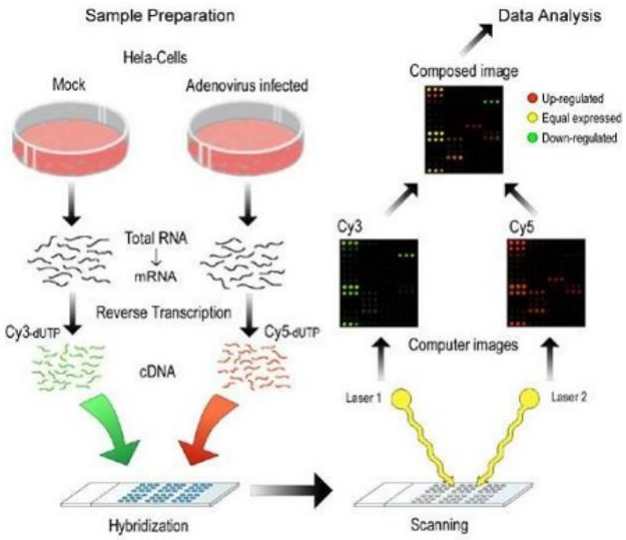


Fig. 1. Microarray Testing Process [1]

only bind with complimentary strands, which transcription produces. A fluorescent color marker is affixed to the cDNA during the transcription process and is used to identify the bound molecules after hybridization. The original DNA strands extracted from the cell lines are discarded, then the transcribed cDNA sequences are exposed to the microarray slide and hybridized. Analysis of the hybridized array is used to determine the presence of reference DNA sequences compared to test RNA sequences, see Figure 1.

Detection of RNA nucleotide sequences in the test material is achieved by analyzing the levels of fluorescent dye in the microarray wells and the associated array coordinates after hybridization. Prior to hybridization RNA is converted into cDNA that binds to the microarray wells. Using this process to measure mRNA concentration indicates the level of protein being expressed by the encoding mRNA. Measuring the quantity of RNA in the test is possible because of the millions of copies of DNA strands affixed to each well of the array and how many cDNA molecules complimentary to the mRNA are found in a well. Concentrations of test cDNA binding to the array substrate is shown in the color expressed from the fluorescent markers attached to the test material. When there is a high concentration of cDNA complimentary to the microarray substrate, the

fluorescent markers will appear in higher concentration in that array location. This is performed following hybridization where the wells on the microarray are stimulated by a laser and scanned by a microarray scanner to determine the amount of fluorescence emitted by each array location [2]. These results can then be quantified and used for further analysis.

2.3 Biological Data Mining

Data mining is a process of discovering patterns and developing knowledge from databases using computational techniques. Data mining has applications for tasks including classification, clustering, association, and regression [4]. These methods fit a model to the data for the purpose of describing or making predictions on the data. Classification is one type of data mining and algorithms can be used for making predictions on the data by mapping the data into predefined classes or variables. Descriptive data mining methods include association analysis, sequence analysis, and clustering [4]. Association analysis produces a rule set that describes data relationships and sequence analysis does the same with additional consideration for sequences of events. Clustering is a descriptive method that groups similar patterns together into cluster groups by comparing the attributes of data records is an unsupervised process requiring no training data.

Clustering is the most popular method used for gene expression analysis and is used to group genes with similar behavior or samples with similar gene expression. There are different cluster techniques used in specific applications including k-means, hierarchical, singular value decomposition, and self-organizing maps [4]. K-means clustering has a faster run time than other methods and is a preferable method when computational resources are limited. K-means requires a predefined number of clusters to be specified and all attributes to be numerical. Numerical attributes are required because associations are determined by computing the distance between records using the numerical attribute values. Prior

to clustering any attributes containing a text value must be converted to numeric values or excluded from the clustering operation.

K-means uses a cluster centroid in each cluster to determine if a record will be associated with a specific cluster group. The centroid holds a value for each record attribute being clustered. Values in the centroid hold the results from calculating the average value of all records in the cluster for that attribute. The final calculated centroid record contains the average value for each attribute in the cluster. Initially k-means creates the specified number of clusters then randomly assigns a record from the data set to each cluster as the centroid values. The remaining records in the data set are assigned to the nearest cluster and the centroid values are recalculated. Following the first round of clustering, the algorithm repeatedly assigns each record in the data set to the nearest cluster and recalculates the centroid until an error threshold is met. The most common error measure check for clustering is done by calculating the sum of squared errors (SSE) [4]. The SSE is calculated by summing the distance from each data point in the cluster to the centroid for all centroids then summing those values, using Equation 1.

$$SSE = \sum_{k=1}^{K} \sum_{i=1}^{a} \sum_{j=1}^{n} (X_{ij} - \overline{X_i})^2, \quad (1)$$

where K = number of clusters, a = number of attributes, and n = number of records. When the SSE no longer decreases from the previous run or decreases by a determined threshold the clustering is complete.

2.4 Visualization

Understanding relationships between expressed genes can provide information on complex functions of the organism and causal relationships in cell processes. Network diagrams are a common method of visualizing relationships uncovered in a biological data mining task on gene expression microarray testing data [6]. Biological visualization networks consist of vertices representing a specific gene or behavior, connected by edges indicating relationships. Algo-

rithmic processes can create these networks if there exists descriptions of the relationships in the data. In genetic identification, and gene expression testing, relationships between data points can be expressed as commonalities or interaction relationships. Commonalities can be discovered by clustering and interactions defined using known relationships between data points.

Generation of a network diagram from biological data can be done by using numerical values in the data, arranging the vertices using a force-directed approach, or a combination of both. Using attribute values as coordinates to plot data can be problematic when the number of numerical attributes is not equal to the number of axes, possibly requiring additional human interaction. Raw data from microarray analysis produces non-normalized data, which must be normalized and scaled to the graph space and may not scale as expected for every graphing application. This approach requires preprocessing and normalization of the data as well as specifying what numerical attributes will be used as axis points. Applying a force-directed algorithm to the nodes uses simulated mechanical forces applied to the nodes to determine the location coordinates in the graph. This is accomplished by simulating charged-particle repulsion between each node (Coulombs Law) and attraction forces between edge connected nodes (Hookes Law) [7]. The process is iterative, each step representing a slice of time where vertex coordinates are recalculated using simulated forces. After a number of iterations an equilibrium is reached and no net change occurs in the node forces completing the process. The result is a graph with uniformly spaced nodes.

2.5 Virtual Reality

2.5.1 Concept

VR is a computer-based system used to produce a realistic environment experience using images, sounds, and head movement tracking. The VR device is composed of a headset that provides stereoscopic images in close proximity

to the user's eyes. Images produced by the headset are displayed on two LCD video screens built into the VR headset or by an external video device such as a smart phone with VR functionality. Head tracking built in to the headset provides feedback to the software, which uses the motion data to control the VR environment view. This allows the user to control the VR environment by rotating their head, simulating a real world viewing experience. VR headsets are primarily used for gaming and game development. The health care and academic industries have more recently begun to experiment with VR devices in their fields as methods for training and medical therapy [8].

2.5.2 Hardware

Development in VR hardware and software implementation is an emerging concept in computer science with growing popularity. There are a number of available consumer VR headsets available with a range of features and capabilities. Headsets are capable of operating tethered to a computer, using its processing power and graphics capabilities, or mobile, using the processing power and video display of a mobile device. VR hardware technology is growing with the addition of eye-tracking hardware, wireless hand tracking devices, room sensing technology, and continuous mobile integration. Tethered operation with a high-powered graphics card and CPU is preferred for testing a VR application with simulation aspects and high computational complexity. For demonstration of the technology on a reduced scale the mobile variants are capable of running simulations and data mining tasks. Because of cost and availability, Samsung Gear VR powered by Oculus is the headset used for this project and is tested on an Android Note 5 as an Oculus signed Android Package Kit (APK) application.

2.5.2.1 Samsung Gear VR

The Samsung Gear VR powered by Oculus is a mobile VR headset supporting operation with VR specified Samsung mobile devices. The headset connects

to a mobile device through a USB 3.0 conection and locks the device into the headset with the screen facing two stereoscopic lenses. The lenses are seen from the inside of the headset when in use and sit less than one inch from the users eyes to simulate a wide range of view. Applications executed on mobile devices divide the screen vertically down the center to provide a single image for each eye. Processing and program execution is performed by the mobile device including all graphics rendering for VR. Images need to be rendered twice on one screen for stereoscopic display and because of Android has limited processing power, additional optimization steps are required during development. Some methods for increasing performance include reducing resolution and limiting frame rates.

2.5.3 Software

Applications for VR headsets can be implemented using 3D game development software such as Unity and Blender. These development environments are maintained and optimized for 3D vector graphics processing and provide a platform ready for VR development. Unity is a highly supported commercial software product targeted towards game development with large communities of users providing support and add-ons. The platform is available at no cost if it is being used to develop non-commercial applications. Because these products are used for commercial development, there are more features and support available for the software than open source products. Unity provides an environment to create 3D graphical programs with animations and scripting capabilities. The environment uses a view with a scene where game objects can be dropped and modified with custom or included scripts. Unity provides built-in build options for many devices including Android VR, Windows, Linux, and Mac OSX allowing fast deployment on many platforms.

2.5.4 Android VR Software

The Android development kit is a software bundle that provides access to a mobile device or mobile device emulator for installing, executing, and debugging applications. The kit is required for a number of operations in the VR application build, mainly to compile the APK, which is the Android application installer file format. Prior to deploying a VR application to a device that will use the Gear VR, a signature file specific to a single device must be requested from Oculus and included with the program source files. The Android development kit is able to retrieve the device identity from the mobile device used to request the signature file from Oculus, which limits testing of the application to only on that device. At the time Unity builds and deploys the VR application to a device, the Android development kit is used to make, install, and run the APK.

The Oculus Development Kit for Gear VR provides the API used in Unity to interact with the VR headset and the Android device. VR applications require information from the headset about head rotation and location, which the Oculus API provides. The Oculus kit includes prefabricated game objects for Unity that have built in functionality for head tracking, which acts as a head controlled camera for the application. The Oculus Development Kit and API are required for any application using a Gear VR headset and the program will not be allowed to execute without the Oculus signature file embedded in the source files.

Blender is an open source development environment similar to Unity and Unreal that provides more options for implementing and modifying vertex mesh objects. Blender's default environment is used for modifying mesh objects, requiring additional configuration and software to develop VR applications. Mesh files created in Blender can be directly imported into Unity or Unreal, which do not have the same mesh creating capabilities. Because of this, developers may use Blender for game object creation of complex meshes that are used by Unity or Unreal.

2.6 Data Simulation

2.6.1 Discrete Model

Computational simulation is a method of gaining insight into a system by reproducing its behavior. A model of the system is created using mathematical formulas that provide output based on simulation input. Computational simulations can provide large quantities of data from a model by performing many calculations that would not be practical without a computer. Discrete simulations model a system, where events occur at specific points in time representing a change in state. The discrete model only considers events at specific points and assumes no change between those points. Discrete simulations can be used with biological testing to determine the accuracy of a model used to analyze the data. Microarray data is synthesized using a mathematical model that creates records with attribute values similar to those found in testing. This allows for a broader range of values to be tested than would be found in a single microarray, which can help determine the validity of a model [9].

2.6.2 Cluster Model

Clustering, an unsupervised data mining method, is used to group unknown data into meaningful patterns. Clustering microarray data with known values can provide a model for determining similarities when used as a training set for simulated record values. Microarray gene expression testing provides numerical values based on the intensity of illumination scanned from each well of the microarray. When this data is clustered, simulated value datapoints introduced to the clustering algorithm will indicate similarity with other records by the clusters they associate to. This type of simulation allows for pattern discovery of test data, assessment of simulated data sets, or point similarity of simulated data points with biologically plausible values. This can give insight into data that

is unknown, undiscovered, or not naturally occurring and provide information about data with similar values.

3 Related Work

3.1 Genomic Database Visual Mining

Microarray expression profiling has seen capabilities expand over several years, initially giving test results in the hundreds and now newer methods providing thousands of results. Growing data sizes can be difficult to visualize and interpret, which can be addressed by using the three-dimensional capabilities of virtual reality. *Genome3DExplorer* is a software solution that utilizes the virtual reality platform to visualize data and bring interaction to analysis [10]. The software uses a specific graph language where nodes represent focused biological objects of interest of the genomic database and edges represent relationships.

The user has a choice of how nodes and edges are displayed to the user by mapping values to the visualizations. Nodes or edges can be mapped by color, shape, size, and transparency, allowing the user many options on how to identify datapoint values visually. This approach is beneficial for large datasets to quickly identify nodes of interest in the scene for further analysis. Because of the limited computational abilities of the Android platform, simple color mapping is used in the project to represent data node values and edges.

A challenge of graph drawing in three dimensions is how to orient the points in the scene with edges influencing the layout. One proposed method of drawing node edges is to assign the length based on a numerical value associated with the edge. This is abandoned because of unsolvable geometrical constraints with certain edge length combinations. A favorable layout approach used in *Genome3DExplorer* is the force-directed placement method, simulating repulsion and attraction forces between nodes [10]. This placement method uses both simulated force and repulsion of randomly distributed nodes for orientation. Each edge drawn between nodes has an associated value, which becomes the intended length of the edge. This attracts the nodes closely to that length while maintaining repulsion forces and distribution. While the *Genome3DExplorer*

project abandoned the repulsive force portion of the algorithm because of performance constraints, both repulsion and attraction are used in this project. The force directed algorithms used in this project are a combination of Unity's physics engine and impulse force algorithms executed each frame.

4 Methods

4.1 Tools

VR development on a mobile device requires a combination of multiple development environments, Android platform tools, mesh graphic design programs, and APK verification from Oculus. Oculus requires all applications that are developed for Oculus hardware to have an embedded key. This key is freely provided to developers to test on a single device, while additional verifications are required to distribute the software publicly. Oculus also provides software that adds functionality to Unity, which is required to interface with the VR headset hardware. Android provides a developer kit that packages and executes applications on the Android device. Unity uses the Android developer kit tools from within the program to compile and execute software directly to the device.

4.1.1 Unity

Unity 3D is a game engine software used to create video games and simulations across different platforms. Programs built in Unity can be designed for computer, mobile device, or tablet use by setting the build type and downloading the required packages. Unity is free for non-commercial use and comes with an asset store that provides access to thousands of free and paid packages for development. The development interface is composed of a layout that includes a scene preview, a scene builder with scene objects, a directory with assets and scripts, and an inspector window for the currently selected object. The graphical scene builder shows a 3D grid, which game objects are placed on and positioned using X, Y, and Z coordinates.

Game objects are controllers that are placed in the scene builder containing the scripts and information about the appearance and activity of an object. All game objects in a scene are updated once every frame during play mode, including the redrawing of an attached vertex mesh or associated display mate-

rials. Game object behavior is determined by C# scripts attached to the game objects, which execute an update method each time the parent game object is updated. Basic game object vertex meshes are included with Unity and many more are available on the asset store. Complex game object meshes that are not included with Unity can be created in mesh design programs such as Blender. Blender can be used to create animations, form vertex meshes, and apply textures to the surface to make highly detailed 3D objects directly importable by Unity.

Game object scripts are created in C# and can be used to configure almost any aspect of an activity in Unity. Because each game object is updated every frame, tasks with high computational complexity need to be distributed to multiple game objects to prevent frame pausing during play. Methods used for simulation aspects of the program such as cluster recalculations can cost many iterations and calculations over a dataset. Unity will wait for a game object to complete its update before any new updates can take place, causing a need for decentralized processing of complex tasks. This can be achieved by many game objects making small calculations on data rather than the centralized approach typically used when clustering data.

4.1.2 Blender

Blender software is used to create the meshes that are rendered in Unity during play. Each mesh is a collection of vertices with their associated coordinates, edges, faces, and material information. Blender provides the capability to create these objects in a visual environment and performs all necessary calculations to rotate, scale, shape, and texture a mesh. Mesh templates are included with Blender and can be used as a base to form a new object. A simple cube mesh contains eight vertex points for each corner, 12 edges connecting the corners, and six faces. Each face can have an image or texture mapped to the individual surfaces, or many surfaces in more complex objects such as the sphere. Sim-

ple meshes with fewer textures perform better in a game environment because mesh components are recalculated and redrawn every frame. The number of centroids used in clustering is typically far fewer than the number of datapoints being clustered. Because of this, spheres are used to represent centroids and cubes are used to represent numerous data points in the scene.

4.1.3 Oculus and Android

The Oculus Development Kit provides the tools needed for the application to communicate with the VR headset and receive head tracking data. Programs designed in Unity for VR use a camera object to determine the display area of the program. The camera has a range of view with a set aspect ratio and displays the scene according to the direction of the camera object. The Oculus Development Kit provides a prefabricated Unity game object camera with scripts attached that control the objects look direction in response to the VR head tracking data.

The Android Development Kit provides much of the functionality needed to compile and deploy a Unity VR application. A Unity project that is set to build for Android VR provides options specific to compiling. Android installers require a manifest that indicates the permissions required for the application and presents the user with options to grant those when the application is first executed. Unity provides a location to store Android specific files on an Android build where the manifest file is placed, as well as the Oculus signature files. Those files are compiled into the final APK and are accessed by the mobile devices as needed.

4.2 User Interface

The user interface consists of all the items the user interacts with in the program. The scene describes the area where visual objects are plotted and rendered, then displayed to the user during program runtime. Visualization is supplemented by movement functionality that allows the user access to any point

in the scene. Menus are presented to the user upon request based on the current activity in the scene and allow control over file loading settings, cluster operations, modeling parameters, and rendering settings. The menus allow the user to control what type of analysis to perform by selecting the types of modeling and visualization to use.

4.2.1 Scene

The simulation user interface is intended for VR and is rendered as an explorable 3D visual environment. The display appears from the perspective of the center of a large cube called the skybox, simulating the view from the center of a large room. The skybox is a set of six images that make up each face of the skybox interior. The distance to the skybox remains static as the user changes position in the environmet while the view rotates with user head movement. The head tracking function of the VR headset provides feedback to the program to adjust the view perspective with head movements.

Game object light reflection is static for all objects in the scene reducing calculations for rendering shadows and light perspectives. User controls are activated using a persistent reticle in the center of the view area by moving it over a user interface (UI) object and activating it with a key press. There are two primary scenes that are used during play mode: one for initially loading the files, and one for displaying and simulating data. The initial scene is an interface for loading the data files and the data explorer scene for exploring clusters, associations, and simulated data.

4.2.1.1 Movement

Movement in the environment changes the VR camera position and location in relation to the rendered game objects. The default controls for the simulation are configured for either a generic game pad or the Oculus Gear VR touchpad. The keyboard controls change the view rotation according to

the movement of the mouse and move the camera location using the arrow key block. The VR headset will change the camera rotation when connected and uses the keyboard or game pad controls for movement and making selections.

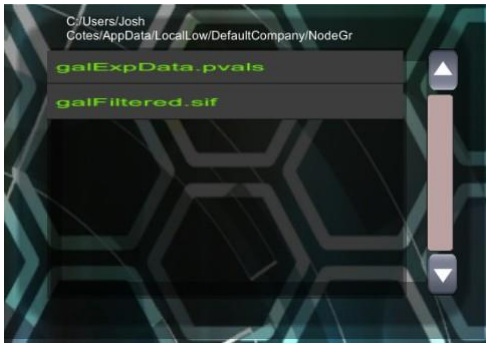
4.2.1.2 Initial Scene Load Files

At program start up, the user is presented with a file explorer window listing all files in the program default directory. The file explorer is represented by a list of game objects containing file names inside a scrolling rectangle game object, providing the ability to scroll through the files and make a selection as seen in Figure 2a. After a file is selected, the user selects if it is a data file or an association file and whether the file has a row of attribute names as seen in Figure 2b. When a data file is loaded, an additional scrolling view is presented displaying objects containing information and selections for each attribute as seen in Figure 2c.

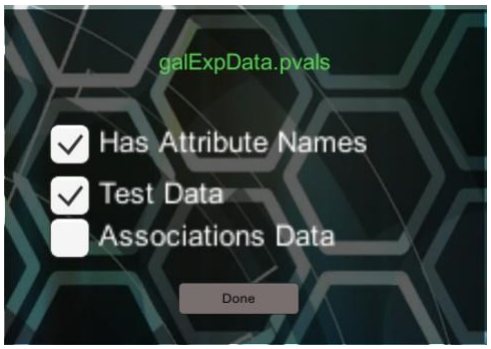
Each attribute has the option to be labeled as an ID attribute, and to be ignored during clustering. Information about the attribute includes the attribute name, whether it is numeric or non-numeric, and the first attribute set as the ID by default. When an association file is loaded, the expected file format has three columns, no names, and needs no additional settings. Two columns contain protein or gene names, and the middle column contains the type of association. After a file of either type is loaded, the file information is displayed on a persistent heads up display ribbon on the top of the display area and the user is given the option to load or reload another file. Both an association file and a data file can be loaded and used in the program. After file loading is complete the data viewer can be entered, launching the scene for data analysis and simulation.

4.2.1.3 Data Viewing and Scene Settings

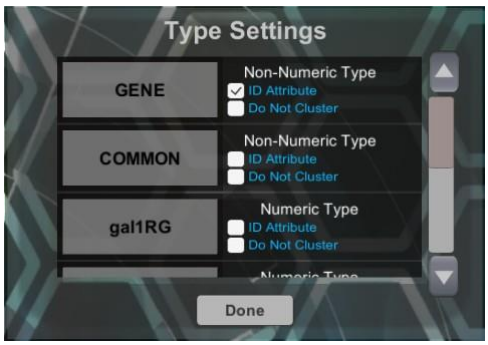
The data viewing area displays a menu with three panes including options for simulating data, simulating associations, and data viewing options, as seen



(a) File Selection



(b) File Options



(c) Attribute Options

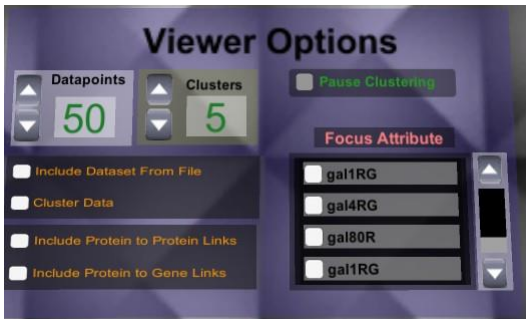
Fig. 2. Load File Options

in Figure 3. The create association and create datapoint menus provide an interface for creating simulated data points and associations in the data viewer. The center viewer options panel contains the settings for displaying data points and associations from file, which can be adjusted for video performance.

The data view settings menus can be hidden or displayed at any time while in the data viewer scene, allowing the user to see immediate updates as settings are changed. Rendered data points and cluster centroids are oriented in the scene using a force directed graphing algorithm to evenly space the objects in 3D space. Records are attached to centroid objects in the scene using a link object, increasing the simulated attraction between the linked points causing clusters to be grouped closer together. The reticle is used to make selections in the menus and provides information about data on the persistent heads up display when hovered over game objects, as depicted in Figure 4.

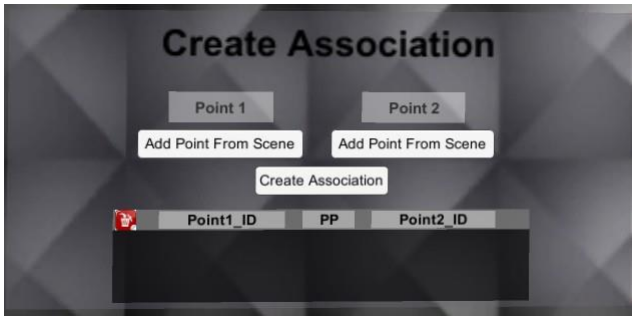
4.2.1.3.1 Viewer Options

The viewer options pane controls how data points and associations from file are handled and rendered, (see Figure 3). The points are instantiated as game objects with a cube mesh in the order that they are loaded from the file, then the game object script associated with the point is passed the record information. The data set counter windows sets how many points will be rendered from the data file when the “Render Datapoints From File” option is checked. This control is necessary because the rendering and clustering operations are computationally complex, causing frame rates to decrease in play mode when all points are rendered from a large data file. This control is separate from the created data points, which are managed from the “Create Datapoint” pane. The count is updated as the value is changed causing new data point objects to be added or destroyed to match the data points count.



(a) Viewer Options

(b) Create Datapoint



(c) Create Association

Fig. 3. Data Explorer Options

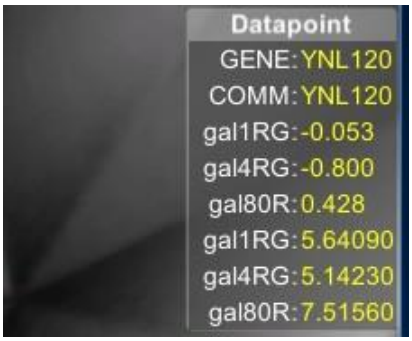


Fig. 4. Data Information

4.2.1.3.1.1 Focus Attribute

Focus attribute selection sets the attribute that all of the records will indicate the value of. The attribute focus is represented by changing the color of the record game object cube using a gradient. All attribute minimum and maximum values are stored during the initial load file process and used to set a color gradient between neutral for zero value and minimum negative or maximum positive values. The neutral color is white, positive color is green for protein induction, and negative color is red for repression. When the focus attribute is set, each datapoint in the scene will change color based on the value that datapoint holds for the focus attribute. The closer an attribute is to positive or negative, the more green or red the game object cube will appear.

4.2.1.3.1.2 Cluster Options

Clusters are instantiated based on the number of clusters selected in the viewer options pane. When the “Cluster Data” option is checked, each cluster is instantiated as a game object with a sphere mesh and communicates with the centroids in each cluster to perform clustering operations. The cluster count can be adjusted during active clustering causing cluster objects to be instantiated or destroyed to match the count. Records are updated with new cluster information as the number changes and records immediately find a new cluster if the associated one is destroyed. There is no check for convergence in the algorithm causing record game objects to continue to re-cluster as long as the “Cluster Data” is checked or until the “Pause Clustering” options is checked. Convergence checks are not considered due to the high time complexity of the calculations and the benefit of constant cluster interaction visualizations to the user for analysis.

4.2.1.3.1.3 Association Links

Associations are loaded from file in the initial scene and can be used to create links between data points if there is an association for those links in the file. Associations can indicate protein to protein interactions or protein to gene interactions. Protein to protein interactions will render a red link between those proteins in the scene. All protein to protein interactions for rendered data points that are present in the association file will be rendered if the “Include Protein to Protein Links” toggle is checked. The data file contains only protein data so all protein to gene links are created if the points are rendered as game objects. Any data points in the scene with a gene link will create the gene as a new data point game object and assign it a constant color. After a data point creates a gene data point a link is created to the new gene point. The new gene point then searches all other data points for associations to that gene and creates links to the point if associated.

4.2.1.3.2 Create Datapoint

Simulating or creating data points is done from the “Create Datapoint” menu pane, allowing custom values to be set for each attribute in a new point, seen in Figure 3b. Each time the “Generate Point” button is selected, a game object data point is rendered and passed the generated values as the data record. Data points can be generated at random by selecting the “Randomize Attributes” button, which uses random values between the minimum and maximum value for each attribute. Individual attributes can be generated at random or all attributes randomized for the new point. Attributes in the new attribute values window can be randomized individually by selecting the random button attached to that attribute. Each attribute has up and down arrows below the value to increase or decrease the value of the current attribute, which is clamped between minimum and maximum for that attribute. The arrows increase or decrease the value by 5% of the range between minimum and maximum for that attribute. The button

labeled “Get Random Datapoint” will search all of the records in the data file and assign the record values from a random record to the created attribute values.

The initial created attribute values can be set to values from a game object data point in the scene by selecting the “Select Point From Scene” button. This option removes the main menu and allows the user to select a point from the scene by hovering the reticle over that point. The attribute values from the point are shown in the data point info, seen in Figure 4, and are the values that will be used for the created record attributes. After the point to be used is found the values are added to the new record with a key press, which reactivates the main menu and adds the values to the created record attributes. The new data point game object is created and added to the created points list when the “Generate Point” button is selected. Selecting the red trash can icon in the list will remove the data point from the scene.

4.2.1.3.3 Create Association

Creating associations simulates an association that would be found in the association file, seen in Figure 3c. This allows the user to select generated gene or protein data points from the scene and create a new association for those points. Creating a new association requires picking two points from the scene by selecting the “Add Point From Scene” buttons. Selecting the button removes the menu and allows the user to find a point with the reticle, then saves that point as half of the association when a key is pressed. The user can select gene to protein, protein to gene, or protein to protein for the new association. Both data points must be selected before selecting the “Create Association” button, which adds the association to the created association list and a link between the points. Selecting the red trash can icon in the list will remove the association and destroy any links to that association.

4.2.2 Biological Data

4.2.2.1 Protein Expression

Microarray result data is publicly available and compiled by organizations that mine and share data from academic papers. The National Institute for Biotechnology Information (NCBI) is a research organization that participates in research on biochemical problems at the molecular level using mathematical and computational means [11]. The organization hosts a public database called GEO (Gene Expression Omnibus), a repository of biochemical data mined from research papers. Data can be searched by experimental conditions, dataset types, result types, and with advanced options for further search refinement. Protein expression data for testing was used from GEO and includes dataset titles: “Anti-leukemia drug etoposide effect on ecotropic viral integration site 1-overexpressing myeloid cells” [12] and “SRPIN803 small molecule inhibitor of SRPK1 effect on retinal pigment epithelial cell line” [13] with no particular significance other than the type of data.

The datasets contain headers describing testing conditions, data sources, species, and protein type info. Records in the dataset include protein ID, test values, and information about the protein. Because the numeric attribute values and protein ID information is the only information needed for the clustering simulation, all headers and non-test-related information must be removed from the datasets before loading it into the program, leaving only the ID and numeric attributes. If the user wishes to include non-numeric data in the clustering simulation they may chose the option “Do Not Cluster” at load time for those attributes, allowing the user to view the information in the program while not interfering with cluster calculations.

4.2.2.2 Protein Interactions

Protein interaction data provides information about proteins that interact with other proteins in a biological process, which can provide relevant information for protein expression microarray result analysis. Biological protein and genetic interaction information is available publicly and maintained by biological research institutions that perform research and mine data from academic research repositories. The Biological General Repository for Interaction Datasets (BioGRID) is a website that maintains and provides genetic and protein interaction data by accessing over 1,400,000 interactions from multiple academic and government protein interaction meta-databases [14]. BioGRID data is available through a website with advanced query options or a representational state transfer interface (REST) service. The web access and REST queries allows the user to search for proteins using a common name or a synonym and returns a list of results containing all known interaction data for that protein. It includes information about the testing methods, research references, protein or gene name, and associated gene or protein name with synonyms.

The REST service provides a way to programmatically access the association database and can be used to build an association list file for the simulation program input. The BioGRID rest service is searched using a HTML access point and returns a results from the HTML query. The query requires an access key provided by BioGRID to be embedded in the URL with a file path containing keys and variables to modify the search. Association files used for testing with the datasets were built using the REST interface and a parsing program written in Java. The association builder program takes one datafile as an input containing protein test data with protein IDs and the numbered column of the ID attribute. The program sends a query to the REST interface for each ID in the input file and creates a new association if a result matches another protein in the input file. The result is an output of associations that may be used as input to the simulation program in combination with the input dataset.

4.3 Modeling

4.3.1 Attribute Selection

Attribute selection and use is determined by the user at the time of loading a data file. When data file selection is complete, the file is loaded and divided into a list of string arrays containing the record attribute values. The expected file type is a comma-separated, space-separated, or tab-separated file. When the file has an attribute names header row those names are stored and used in the program as identifiers. If no names exist in the file, the program generates attribute names starting with “Attribute 1” . . . “Attribute *n*”. The next line from the file is read as a record and used to infer numeric types from the data by trying to parse each value as a double. Types that do not parse as a double are stored as non-numeric strings and will not be clustered. The clustering calculations in the simulation can only operate on numeric data and all data is assigned an enumerated type of NUMERIC or NON_NUMERIC. Both numeric and non-numeric type attribute data is available in the data viewer when viewing information about data points loaded from file. Non-numeric data is not considered or available when adding new or simulated data points to the simulation.

Association files are expected to contain only three attributes and no names row in the file. The first attribute contains a protein or gene ID, the second attribute contains an association type “pp” for protein to protein or “pg” for protein to gene, and the third attribute contains the associated gene or protein ID. Association files may have an attribute names row and will be used if the user specifies to do so at the time of file loading. Links are formed and game object datapoints are created depending on the association type using green cubes for genes, blue cubes for proteins, red links for associations, and white links for clusters. The associations are mapped to a key-value structure using the first ID as the key and the value as an array containing the second ID and type of association. Associations loaded from file are stored reciprocally and without

duplicates. Each association loaded from the file is stored as (*ID*1*,* [*ID*2*, type*]) and (*ID*2*,* [*ID*1*, type*]), causing every record ID to be represented in the key list.

4.3.2 Record Objects

The data viewer renders the data points in the order they are read from the datafile, each point containing relevant data and attribute values from the corresponding file record. The information held by each record game object include attribute types, minimum and maximum attribute values recorded when the file is loaded, clustering script pointers, attribute names, attribute values, a pointer to the game controller script, and the associations list. The record objects perform many of the clustering and record interactions each frame update reducing the work load on individual objects and maximizing frame rates. The focus attribute selection is monitored by each data object and changed on a one second delay. The attribute in focus is reflected by a linear interpolation between two colors using the attributes percentage to maximum value or percentage to minimum value. If the attribute has a negative value, the object becomes more red on the gradient between white and red. The positive values are represented by a gradient between white and green with green being the maximum value. The focus attribute options have no effect on the object color if a record is not from file or is an associated gene.

4.3.3 Instantiation and Destruction

At the time data points are destroyed, each record object must first destroy any links to centroids and associations and then itself be destroyed. Record instantiation and destruction is controlled by a coordinator script and managed by the game controller. The rendered data point objects are stored in a list data structure and are destroyed in the order of most recently created first. The record count and “Include Dataset From File” option set the destruction or instantiation

of record objects. This is reflected by destroying the objects as the datapoints count is reduced, instantiating records from file when increased, or destroying all records when the toggle is deselected. On record creation a new record game object is instantiated and passed the correlated file data. On record destruction the coordinator removes the last object placed into the records list and calls on the record object script to destroy all links. The record object maintains lists of all links and genes connected to that record ordered by centroid, gene, and protein. When the command to destroy links is issued to the record, any centroid and protein links are immediately destroyed. If any associated gene objects exist, the record queries the gene object for any other records associated with that gene. Only the link to a gene with other associated records is destroyed while a gene with no other associated records is immediately destroyed with the link to it. After all links have been destroyed the coordinator object destroys the record game object and removes it from the record object list.

4.3.4 Clustering

Clustering operations are coordinated using a combination of game object records, centroid objects, the cluster script, and the game control script. Each point has an associated script that stores the record as a string array with an array of enumerated DataTypes indicating the attribute type for each column. The information about the attributes is used primarily during the cluster operations. Each data point game object performs cluster operations every three seconds on the game object frame update cycle. On the frame that a cluster operation is performed by a record object, the time is marked and is not updated again until greater than three seconds have passed. The clustering delay allows all record objects to complete a clustering cycle before the next cycle starts when many data points are rendered therefore reduces computational load.

Each record is associated with a central cluster script object and a centroid game object script. The cluster script keeps track of cluster cycle information,

regulates update cycles, and sets the focus attribute. The centroid script contains the centroid values and coordinates reporting record objects with the cluster script. At the time record game objects are instantiated they are passed a pointer to the cluster script and perform a check at each frame update on variables of the script. The cluster script indicates whether the clustering operations are active or paused and contains a list of centroid object scripts during active clustering. At every frame update the record object queries the cluster script to see if clustering is on and which focus attribute is active then performs the necessary calculations at each time interval.

4.3.5 Perception

Unity uses meters as Units of measure, translating coordinate based scenes into real-world metric dimensions perceived by the user. Clusters and centroid diameters are set to 100 meters and instantiate in the center of the data explorer view at coordinates X=0, Y=0, Z=0 (0, 0, 0) shown in Figure 5. Menus are provided on demand using a key press and appear ten meters in front of the camera. When the menu is active in the data explorer, all movement aside from head rotations is disabled. The menu dimensions are adjusted to fill the screen space providing access to all menu options with minimal head rotations. The camera is set 80 meters away from the center of the data explorer at (0, 0, -80), placing the game object instantiation point in front of the camera.

4.3.6 Force Directed Layout

Game objects and links both determine the forces used to distribute game objects in the scene. Game object scripts perform attractive and repulsive force calculations at each frame update causing the objects to move towards an equal spaced layout over time. The attractive force of game objects adds an impulse force to the object in the direction of the center of the scene coordinates (0, 0, 0). The force algorithm uses the location of the game object to calculate the

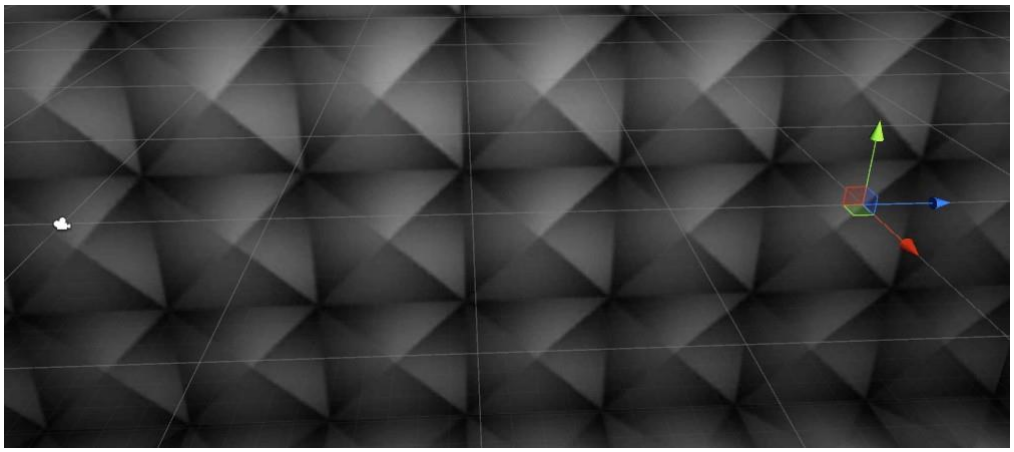


Fig. 5. Unity Scene Showing Camera and 3D Axis Indicator

impulse strength towards the scene center. The formula multiplies the negative values of the normalized location game object vector by the set object mass in kilograms multiplied by a velocity constant, or

$$\text{attraction} = \frac{-\mathbf{X}}{|\mathbf{X}|} * \text{mass} * V, \tag{2}$$

where X = location vector, and V = velocity constant.

The repulsion algorithm determines repulsion force by using a Unity physics sphere around the object and the collisions with other object spheres. This is done using a list of all objects within the sphere radius collision area and applying a repulsive force to those objects. For each object in the sphere collision area, the direction is calculated to the collision object using the difference between both object center vector positions. The repulsive force is calculated using by dividing a repulsion constant by the distance between the objects squared, or

$$\text{repulsion} = \frac{R}{(\overline{PQ})^2} \tag{3}$$

where R = repulsion constant, and $\overline{PQ}$ = distance between points. This reduces the repulsion as distance between points increases.

Repulse and attraction calculations are performed at each frame update and game objects are in constant movement. The force-directed calculations are

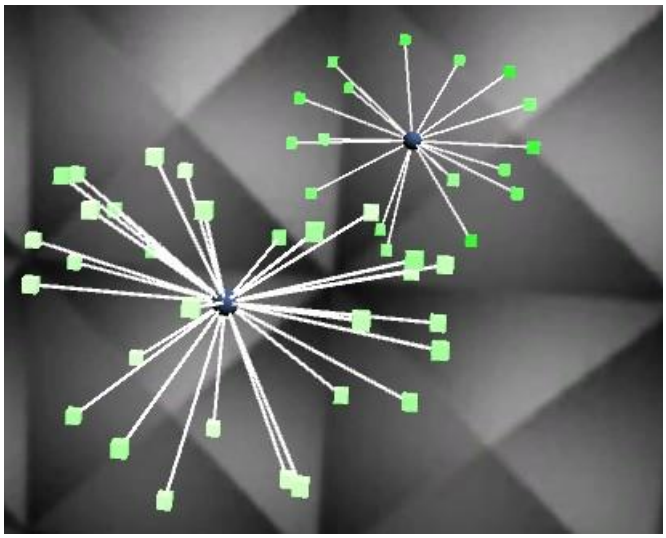


Fig. 6. Clusters in Viewer

computationally complex and are the cause of some performance limitations when rendering large data sets. Attractive forces from link objects provide the grouping for clusters and cause all clusters to be grouped at a distance. Many link objects from record objects to centroid objects cause the records to group more closely and the repulsive force of the objects is combined into a group of repulsion spheres. The links attractive forces cause the clusters to act as one large repulsive object rather than many small objects, seen in Figure 6.

4.3.7 Unsupervised Clustering

4.3.7.1 K-Means

K-means clustering is the base algorithm used for the clustering process in the simulation with some modifications. The clustering operations are on-going and do not stop at convergence allowing the user to see minor cluster changes at all times. Clustering operations are controlled by using the “Pause Clustering” option or turning clustering completely off. This also allows adding simulated data points to a scene with clustered datapoints to determine the pattern similarity with other records in real-time. The clustering algorithm is dynamic and changes as data is added to the simulation. Because clustering operations are

updated for each frame, new records added to the simulation have an immediate influence on the clustering calculations. When clustering is turned on, simulated points are immediately associated with the closest centroid and contribute to the centroid mean values used in re-clustering. Because of this, simulated datapoints affect the outcome of all clustering operations while the simulated point exists.

4.3.7.2 Single Record Data Simulation

Single point data simulation is accomplished by using the “Create Datapoint” menu, as illustrated in Figure 3b, where datapoints are added manually by selecting the values for each attribute. The menu contains an object for each attribute with the attribute name and arrow buttons to increment or decrement the value and a randomize button. Values are initialized for each attribute using pseudo-random numbers in the range of minimum to maximum value for that attribute. The arrows are used for manually setting the value and increment at a scale of 5% of the attribute value range constrained by the attribute minimum and maximum values. The menu contains buttons for randomly assigning values to the attributes, using values from a datapoint in the scene, randomizing attributes values, or using the attribute values of a random record from the data file.

To set the initial values of the attributes from a datapoint in the scene, the reticle is used to find a point and the values are used when selected. When the button is pressed to select a point from the scene, the menu is removed and the reticle is hovered over points in the scene until one is found to use. The information from each datapoint is displayed on the heads up display, as seen in Figure 4, giving a preview of the data that will appear as the simulated attribute values. Once the point is selected, the datapoint attribute values are added to the simulated datapoint and can be modified or instantiated in to the scene.

Randomizing attributes using random numbers generated by the device and

can be applied to attributes using the “Random” button in the simulated datapoint attribute value object or the “Randomize Attributes” button in the menu. Both random buttons use a random number to generate values within the attribute value ranges. The “Random” button within the attribute object applies the random value to only that attribute while the “Randomize Attributes” menu button randomizes all attributes. The “Get Random Datapoint” button selects data values from the loaded records stored in memory. If all file data points are currently instantiated in the scene, the button will randomly select a datapoint from the scene and apply the values to the created datapoint. When there are more datapoints loaded than rendered, the random record selected may be from points in memory that have not yet been rendered or from a point in the scene.

The point is instantiated in the scene by pressing the “Generate Point” button, which immediately generates the game object as a green cube with the generated point data. Each point generated is added to a scrolling list in the create datapoint menu indicating the values and name of the point. Each list item associated with a datapoint allows scrolling horizontally through the attribute values and has a trash can icon to destroy the point from the scene. When destroying the point in the scene all reference to the point is removed from the simulation, then the item is removed from the list.

4.3.7.3 Set Simulation

Simulating the dataset is done by applying preprocessing to the original dataset to extract and combine values with simulated or similar test values in a new file. Simulated datasets can include data from the original file that is selected using specific attribute ranges, IDs, or at random. This data can be combined with simulated data or other data from similar testing using specified record selection to view the cluster groupings. Simulated data sets can be processed and generated with the BioGRID RESTful Association Grabber tool GUI shown in Figure 7, and used to build a list of associations in the file and

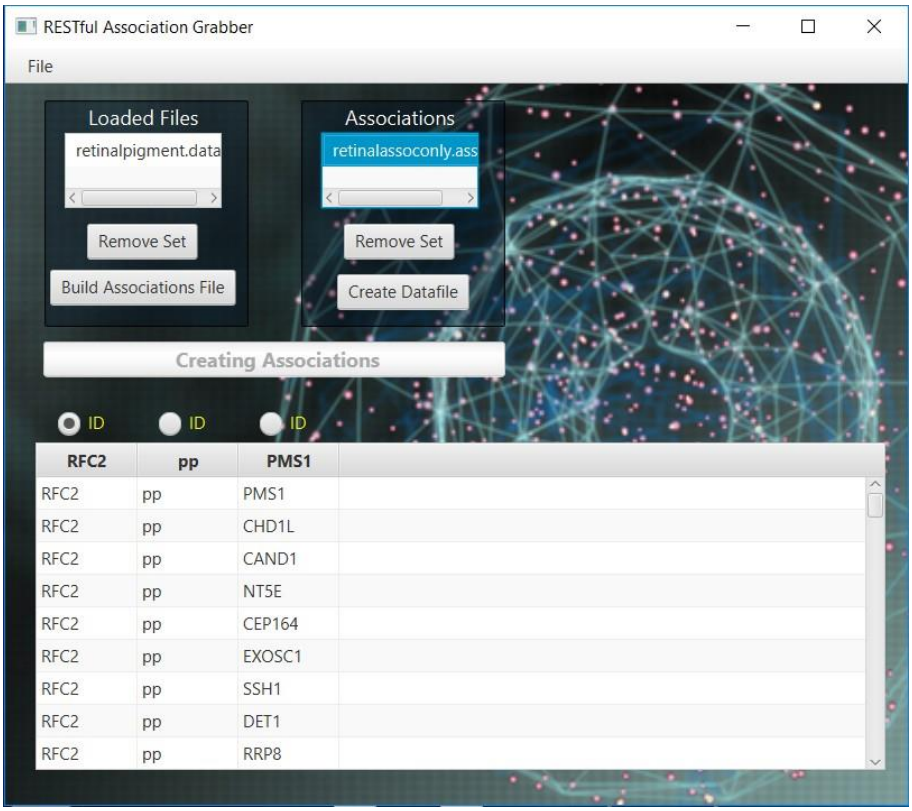


Fig. 7. RESTful Association Grabber

provide further information about the dataset. Simulating and combining sets with the association tools can provide information about overlapping test results and show the interaction relationships not apparent in individual tests.

4.3.8 Data Association

Data association generation is most effectively done using the RESTful Association Grabber tool during data preprocessing. The association grabber searches the BioGRID REST service for all record ID's in the datafile and creates associations for IDs in the file. The BioGRID service contains interactions for more than 1,467,186 proteins and genes providing a solution for large scale association searches [14]. One drawback of the service is the slow query response time can cause data files with 5,000 or more records to take over ten minutes to query and build all associations.

The Association Grabber tool was developed to generate data files contain-

ing associations or to combine data files to a single file for input to the VR simulation program. When two or more datasets are loaded by the association grabber tool, the user has the option to generate an association file, combine all records in the files, or combine data files by associations found. Generating an association file requires selecting an ID column and searching for associations using each ID. Each REST query to the BioGRID server will return a list of all combinations of known protein interactions with an associated gene. This list is used to search the dataset for matching associated IDs and adds the association to the list if a matching ID is found. The resulting association list includes only associations of data found in the original input dataset.

Joining files using the association grabber tool is done by combining all of the raw data in the file or by only including records that have associations in the files. Joining by association will use a premade association list or generate one while joining and only add records with IDs matching those found in the association list. This process results in an association file and a datafile file only containing associated records and will all display with association links in the VR simulation program. When the combine all records option is select for joining files the resulting datafile is a combination of all records from both files.

In the VR application data explorer view, association creation is limited by data points loaded in the scene. The “Create Association” menu provides the interface to create associations in the application, as seen in Figure 3c. This is accomplished by accessing and viewing the menu, then selecting the “Add Point From Scene” button for both points. Selecting this button closes the menu and waits for the user to select a datapoint from the scene. Datapoint selection is done by hovering the reticle over a datapoint to preview the data in the datainfo panel, shown in Figure 4, then selecting the datapoint. After both points have been selected, the “Create Associations Button” will add the association to the list of created associations in the menu and create an association link between the two points in the scene. The associations are deleted using the list of created

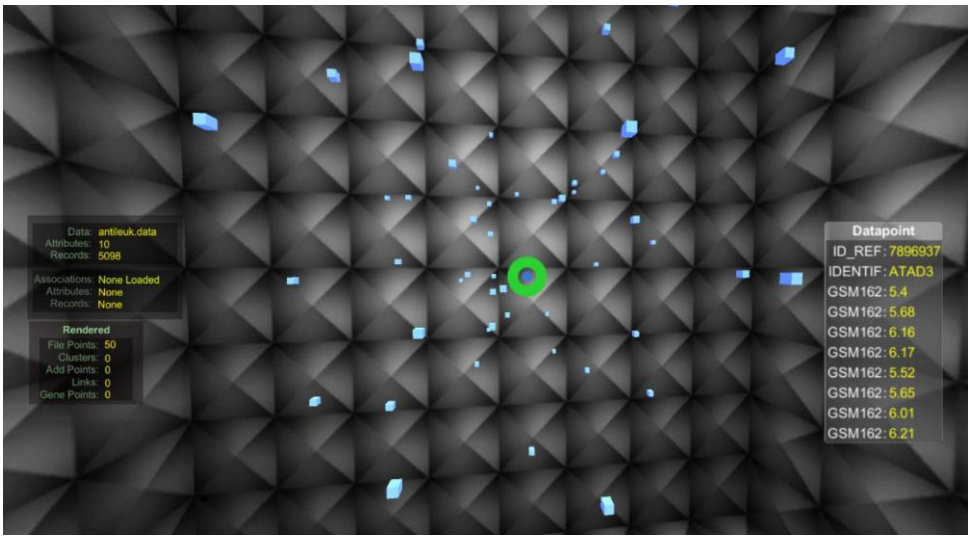


Fig. 8. Data Points From File Arranged Using Force Direction

associations in the menu by selecting the trash can icon on the association to destroy.

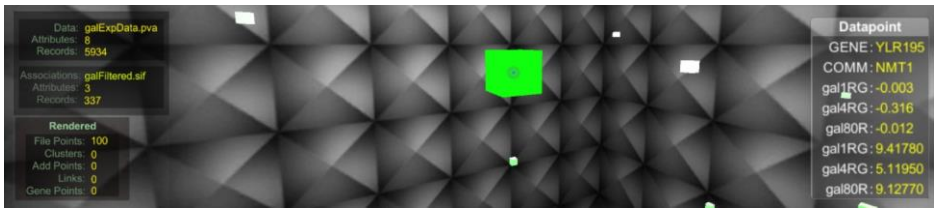
4.4 Visualization

4.4.1 Data Points

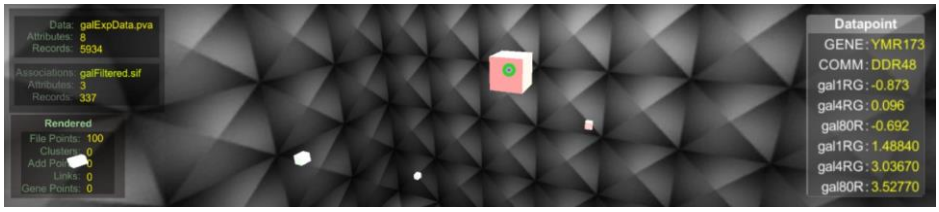
4.4.1.1 File Loaded Data Points

Datapoints exist in four classification categories and include: loaded from file, generated gene by association, generated datapoint in menu, and centroid. Datapoints loaded from file are distinguished by color and focus value. The default appearance of file loaded datapoints are light blue cubes generated at 80 units (meters) in front of the default user display location. As points are instantiated from file they immediately activate the force generated layout process, driving the instantiated points away from other points as illustrated in Figure 8.

The center reticle shown in Figure 8 sets the values in the Datapoint pane, also seen in the figure, each time it is hovered over a point in the scene. Setting the focus point in the data visualization menu will change the datapoint game object color to reflect the attribute value of that point. The game object color is updated on an interval and will check and change the focus value if needed every



(a) Focus "gal1RG" Test Value Induction



(b) Focus "gal1RG" Test Value Repression

Fig. 9. Focus Attribute Showing Induction and Repression

one and a half seconds. Setting the focus for an attribute will display darker red for the level of repression or negative values as seen in Figure 9a, brighter green for induction or positive values as seen in 9b, and white for near zero values.

4.4.1.2 User Generated Data Points

Generating points using the "Generate Datapoint" menu in the scene viewer creates a point in the scene containing the values entered at the time of creation as shown in Figure 3b. Generated points are included in all clustering operations and are clustered the same as points loaded from file. The points are rendered as green cubes in the scene and are moved in the layout using the same directed force as all other data points as shown in Figure 10. When the reticle is hovered over a generated point the data contained in the point is displayed in the Datapoint pane with the ID field showing the generated ID and any non-numeric fields displayed as NonNum. All generated datapoints are deleted through the "Generate Datapoint" menu by finding the datapoint in the scrolling list of created points and selecting the trashcan button.

Datapoints generated in preprocessing loaded by the VR program will be rendered as file loaded points and are not differentiated between other file loaded points. All records loaded from a single data file are assumed to have the same

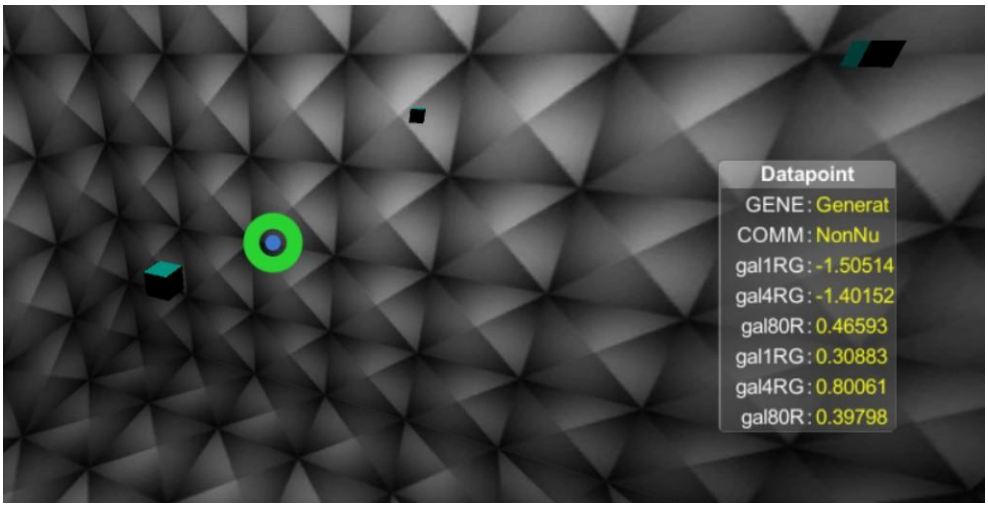


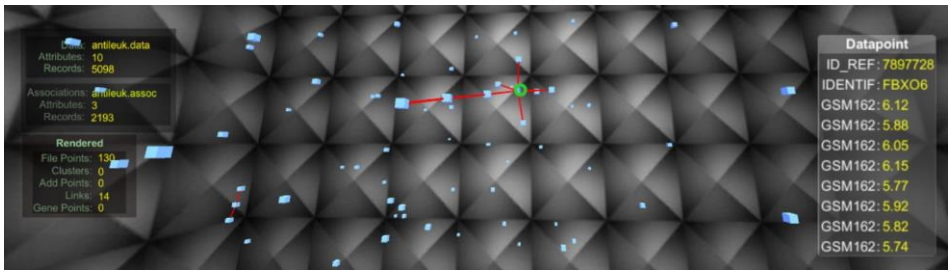
Fig. 10. User Generated Points

attributes and be of the same type. Future work on the RESTful Association Grabber tool and the VR BioAnalyze program would include options to describe the data in the output file for each record during preprocessing, and adding additional handling of files and types to the VR program.

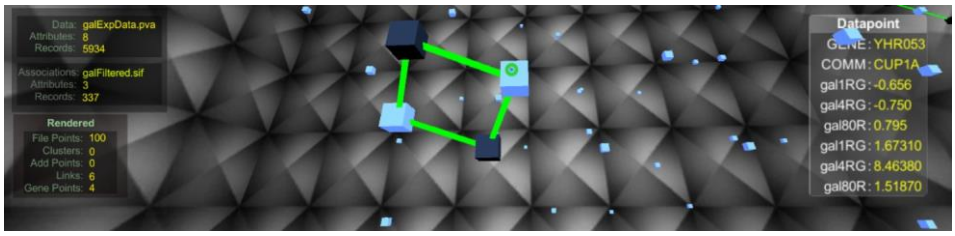
4.4.2 Data Association

Associations are displayed as links rendered as a Unity LineRenderer game object between two points. Association types are distinguished by line color between points and include: green for gene to protein links as seen in Figure 11a, red for protein to protein links as seen in Figure 11b, and blue for user generated links as seen in Figure 11c. Green and red links are generated by loading an association file, which can be generated manually or by using the association grabber tool.

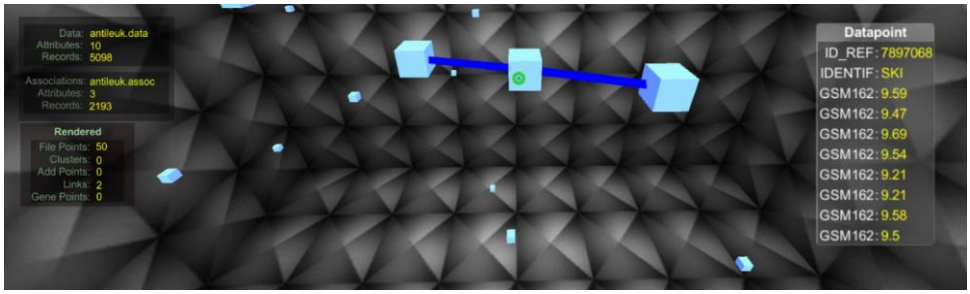
Datapoint game objects for genes are created when protein to gene interactions are discovered in the association file for a point rendered in the scene. Because all file loaded datapoints are assumed to be of protein type, genes are generated using only an ID and only generated when a rendered file loaded datapoint is found to have an associated gene as illustrated in Figure 11b. Generated gene points are rendered as blue cubes and are managed by the associated dat-



(a) Protein to Protein Associations



(b) Protein to Gene Associations



(c) User Generated Associations

Fig. 11. Datapoint Association Links

apoints. At the time a file loaded datapoint game object is destroyed, the point checks all associated genes to see if the gene has any other associated proteins, then destroys the gene game object if there are none. The management of gene datapoints in the scene by file loaded point game objects causes all gene datapoints to be destroyed if they have no associated proteins in the scene.

4.4.3 Clusters

Clusters are formed when the clustering operations are turned on and links are formed to centroid game objects. At the time clustering is turned on, a number of centroid game objects are rendered as blue spheres and the datapoint game object scripts begin clustering and instantiating links to centroid objects as shown in Figure 12. The datapoint to centroid links are rendered as white

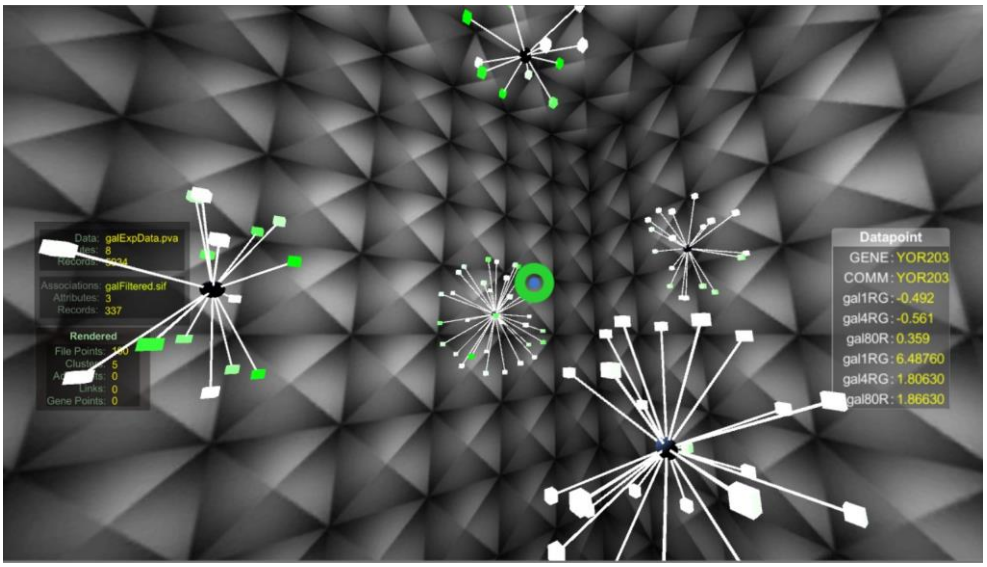


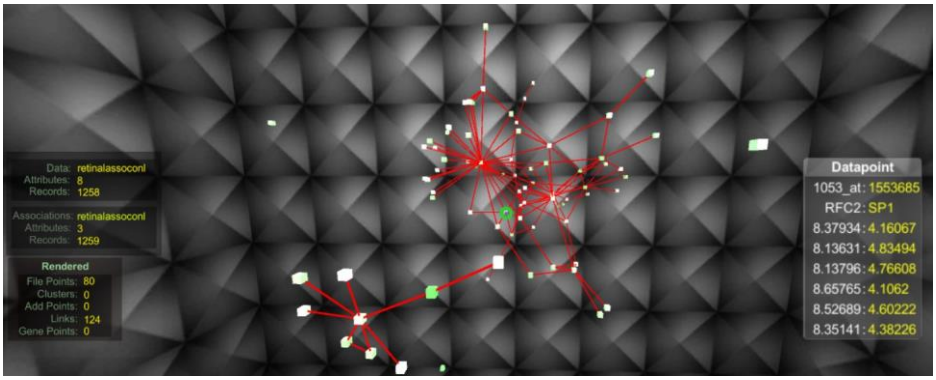
Fig. 12. Datapoint Links to Centroids Forming Clusters

line objects and positively affect the attractive forces of the datapoint objects at each end of the line. The increased attractive forces cause the datapoint objects attached to a centroid object to gather closely around the centroid and move as one larger object.

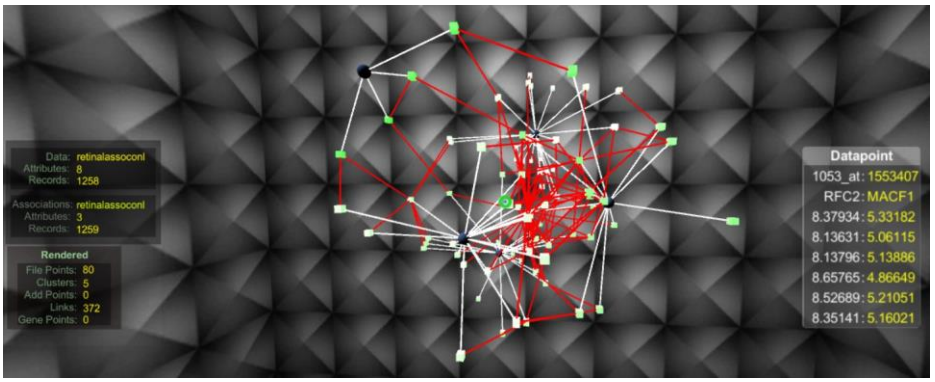
While clustering is active in the scene, the datapoint objects re-cluster and find the closest centroid every three seconds and do not stop at any convergence point. Each occurence of a datapoint finding a new centroid leads to the destroying and instantiating a centroid link. When a datapoint destroys its active link to a centroid and instantiates a link to a new centroid, the link is formed and the datapoint is directed by the additional attractive forces of the link to the new centroid. This active type of on-line clustering allows new points to be introduced to the scene, which immediately form links to the closest centroid and influence the outcome of all clustering operations.

Activating clustering will result in different initial centroid seeds depending on if datapoint game objects are currently active in the scene or datapoints are added after the centroids. When centroids are added to the scene by turning clustering on and settings the *k* number of centroids before any datapoint objects are rendered, the first *k* rendered datapoint objects will become the initial

centroid values. When datapoint objects are active before clustering is turned on, the initial centroid values will depend on the datapoint update order during a frame cycle. The resulting centroid initial values will be the first datapoint to update after clustering turns on, followed by the next $k - 1$ updated datapoint game objects.



(a) Association Mapping Datapoints



(b) Association Mapping Clustered Datapoints

Fig. 13. Association Mapped *Retinal Pigment Dataset*

4.4.4 Data Association Mapping

The RESTful Association Grabber tool can be used to create a data file containing only records that are associated with other records in that file. When the tool is used to build an association list or one is loaded from file, that list is compared against the records from a loaded data file and a new data file is generated.

The resulting files can be saved and loaded in combination with the association file in the VR program. The associations can be seen by activating the ”Protein to Protein Associations” in the viewer settings, which connects all associated datapoint game objects by red line objects. Each datapoint in the file will have one or more associations with records in the same file and will be rendered as datapoint game objects interconnected by red lines as illustrated in Figure 13a.

Using association-only data can be used in combination with clustering operations in the VR program, which will render centroid links, protein to protein links, centroids, and datapoints. Clustering in combination with association-only data and focus attributes can provide additional insights into patterns found when proteins are associated in a gene expression process as shown in Figure 13b.

5 Results

Testing on the data was performed by analyzing datasets found on NCBI Geo and associations found using the RESTful Association Grabber tool. The tests explore the functionality of cluster and association modeling in combination with focus attribute settings. The platform is tested for performance and rendering capability by introducing datapoints into the scene and monitoring frame rates.

5.1 Exploring and Viewing Data

The VR environment is fully utilized by implementing movement in the 3D environment, allowing for 3D placement and exploration of datapoints. Datapoints represented in a two dimensional environment have the limitation of a single coordinate plane and can become unintelligible when many points share the same coordinate space as shown in Figure 14. The 3D environment adds a viewable dimension where many more datapoints can be displayed and points are distributed so any individual point can be seen using movement.

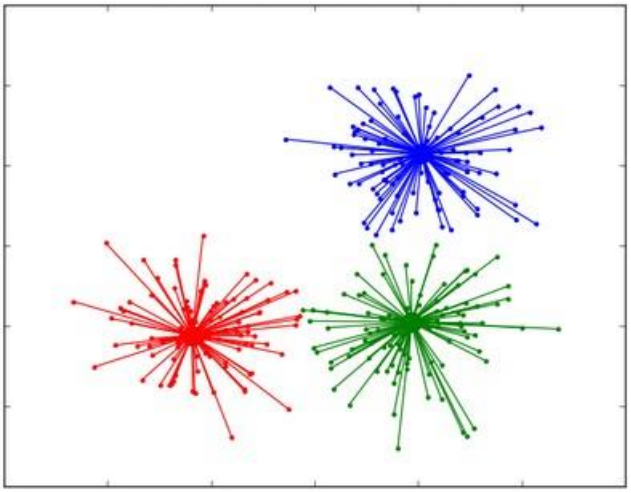


Fig. 14. 2D Visual Cluster Output.
[15]

The ”Datapoint” menu shown in Figure 4 displays the attribute information about the closest visible point when the reticle passes over the point in the scene. Where a two dimensional graph would obscure points in a similar space, the three dimensional environment provides options for changing the orientation to view any datapoint in the scene. Movement allows the user to maneuver the camera position in the scene to bring any datapoint into view of the reticle.

5.2 Cluster Analysis

Three-dimensional cluster viewing in virtual reality provides a unique way to gather information about clusters by providing access to each datapoint involved in the clustering operations. Combining attribute focus with clustering operations provides a visual representation of attribute values and can indicate common values associated with grouping patterns. When clustering is active, the movement system allows the viewer to rotate or move around datapoints to expose other points that may be hidden.

A limiting factor of the Android VR platform is the computing power available to perform clustering and render the scene, which is apparent when the number of rendered points is set above roughly 55 with clustering activated. Active clustering with more datapoints rendered in the scene causes the rendering frame rate to drop and the scene can flicker and jump frames leading to VR sickness. When fewer datapoints are rendered, clusters form and are oriented in the scene using force direction and maintain at least 30 frames per second, providing an experience without frame jerking or flickering.

The *Retinal Pigment Dataset* is a set of microarray results discovered during research of *SRPIN803 small molecule inhibitor of SRPK1 effect on retinal pigment epithelial cell line* [16] and found using a protein expression search on NCIB GEO Datasets [11]. The set contains records pertaining to specific proteins with attributes indicating the reactivity to bait-protein microarray markers used to determine the expression level of the protein. Setting a focus changes the

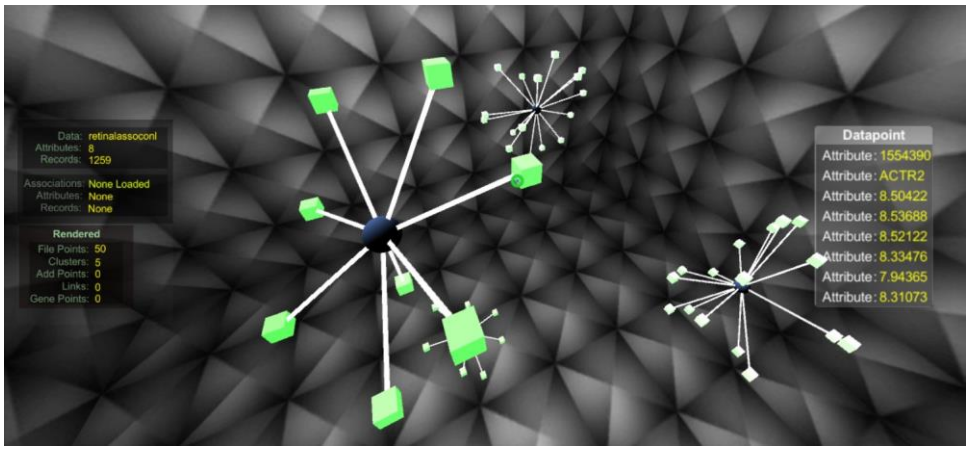
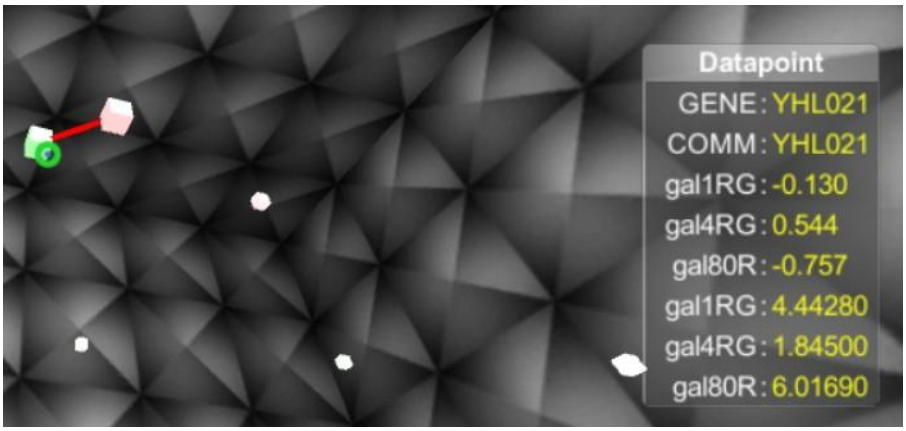


Fig. 15. Retinal Pigment Dataset Clustered With Focus Attribute 3

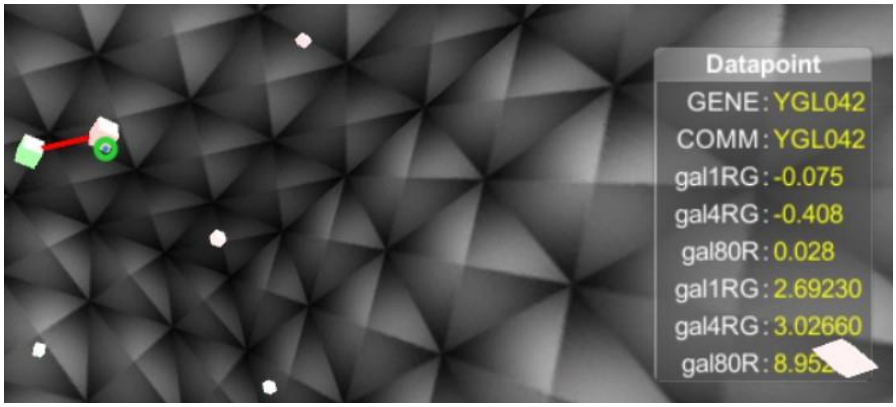
color of the datapoint game objects in the scene and allows the user to quickly gain insight about attribute values grouped in clusters. When an interesting datapoint or cluster is found, movement can be used to bring the objects fully into view. Analysis on the Retinal Pigment dataset using these techniques can quickly show the quantity and identification of the proteins expressed under the test conditions as shown in Figure 15. The number of datapoints rendered is set to 50, the number of clusters is set to five, and the focus attribute is attribute three. The result shows clusters that have grouped datapoints by expression level, which is confirmed by hovering the reticle over the points and viewing the attribute values.

5.3 Association Linking

Showing association links in combination with focus attributes provides a method to gain information about proteins acting together during gene expression activity. When using association mapped data files, the number of links will limit the number of points that can be rendered in the scene. The association links perform the same force directed operations as the cluster links, slowing frame rates when many links are rendered in the scene. Unlike clustering, more points and links can be rendered in the scene without the clustering operations performed simultaneously. The Android VR headset will render up



(a) Protein to Protein Association Focus Attribute *gal4RG* - Left Point



(b) Protein to Protein Association Focus Attribute *gal4RG* - Right Point

Fig. 16. Protein to Protein Association Links With Focus

to roughly 70 datapoints and about 70 association links before frame rates drop below 30 and the reduced frame rate becomes visible to the user.

Association link testing was performed using a dataset of microarray results indicating protein expression levels present during gene activation. The *Disulfovibrio Vulgaris mRNA Expression Set* is a collection of records containing attributes indicating the expression level of three mRNA proteins for 5934 genes at time of gene expression under normal and perturbed growth conditions [17]. The association data built for the set is compiled and distributed with Cytoscape, a biological data mapping software, containing both protein to protein and protein to gene interaction data for Disulfovibrio Vulgaras. When the two datafiles are loaded in to the VR Simulation program, activating all association links while using focus attributes can give an indication of how co-expressed mRNA

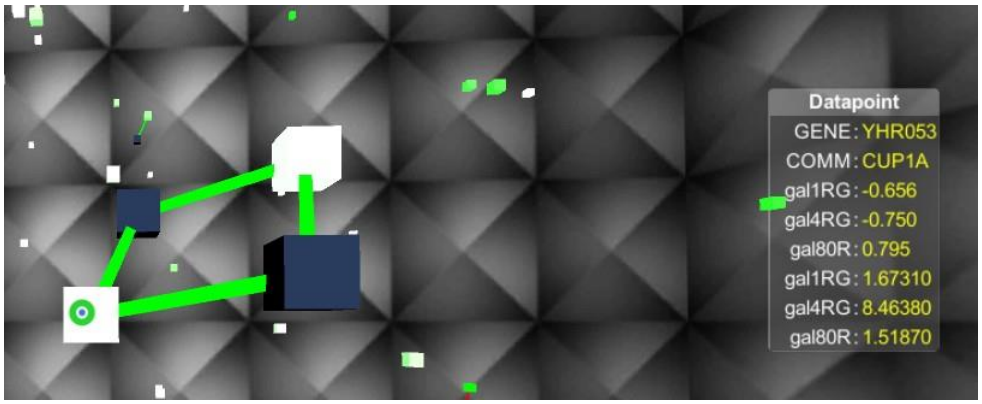


Fig. 17. Gene to Gene Association Focus Attribute *gal1RG*

proteins affect gene expression. Shown in Figure 16] is the result of rendering 100 points from the dataset with protein to protein association links turned on and focus set to the *gal4RG* attribute. The link can indicate that the two genes are possibly co-dependent gene expression or share expressed protein dependencies. The focus attribute indicates the repression or inductiction levels for *gal4RG* showing that during*YHL021* expression, *gal4RG* mRNA is expressed and during *YGL042* expression, *gal4RG* is repressed.

Gene to gene association links provide an additional indication of genes that are co-expressed, showing genes that are not in the dataset known to be associated during gene expression. Using the datasets for the previous test, the results show that the linked genes may associate with genes not in the set during gene expression as shown in Figure 17. The focus attribute is set on *gal1RG* and indicates there is slight repression of the focus mRNA when both genes in the dataset are expressed. This can provide further information about the activity of mRNA and gene expression that may be unknown during the expression profiling experiment.

5.4 Summary

The motivation behind this project was to implement a three-dimensional and VR-explorable data visualization tool. The goal is to provide an intuitive

and inclusive way to understand data that allows for viewing all points, which is not always possible with a two dimensional data representation. This is accomplished using the combination of tools implemented in the VR program including where and how datapoints are rendered, focus attribute colorization, association linking, and cluster grouping. Using the functions of the VR program in combination with each other provides different methods to analyze the data.

The initial enabling and rendering of file datapoints shows the user an overview of the data being rendered. When “Include Datapoints From File” is turned on, a point is rendered once per frame in the center of the scene then distributed by the force directed algorithm used by the points. This gives the appearance of the points appearing in the center of the scene then moving to relatively equally spaced locations. Rendering datapoints before or after activating associations or focus attributes has a different affect on how points are rendered in the scene. When focus is set first, the attributes are rendered with their focus attribute color and give an initial interpretation of attribute values as they are instantiated.

Setting the association links to active prior to rendering points will immediately link the points as they are instantiated in the scene if the points have an association. The association links forming at instantiation causes the datapoints to have an immediate connection and will keep those points attracted together much stronger than those without links. Links formed at time of instantiation have the added benefit of showing the user how many links and what types of links are rendered before they are distributed by force direction.

The RESTful Association Grabber tool provides a way to gather information about protein associations from BioGRID REST service on all data points in a file. BioGRID provides a website interface to look up individual proteins and return a result of all associated proteins from their service, which can be useful when the number of proteins to check is low but not on a large dataset. The association grabber tool can provide association data on a much larger scale

than the website interface and is useful for finding all known associated proteins within a file. This method was used to build all association lists used for testing with the exception of the *Disulfovibrio Vulgaris mRNA Expression Set*, obtained from the Cytoscape test data website.

The combination of analysis tools in the VR program allow the user to analyze data from many different perspectives. Combining clustering with association linking provides the visualization information necessary to determine if linked datapoints have closely related test data. Setting focus points in addition to clustering and associations shows the relationship between attributes, associations, and attribute value similarity, providing additional information not available in a two dimensional data representation.

6 Future Work

There are a number of available and researched data mining algorithms that are appropriate for different data mining tasks, including different clustering algorithms and various preprocessing operations. The use of k-means was explored for this project primarily due to the performance and ease of adaptability to the frame-by-frame update system of unity as well as it's ability to quickly group datapoints in an unsupervised environment. One method for clustering genetic data is by using targeted gene clustering to determine relationships to specific datapoints [18]. This requires prior knowledge of the data and points of interest to be declared and considered in clustering operations which is. out of the scope of this project.

The user interface was intended to provide information about the scene, datapoint, loaded files, and program operations. Additional comparison methods would be useful to select datapoint and view the attribute values side by side and the ability to select specific datapoints from file to render or compare would provide valuable analysis functionality. These functions are left out of the scene due to the performance limitations of the Android device. Rendering lists of objects containing information about file datapoints as well as tracking multiple comparison points adds additional frame update objects, slowing rendering. A tethered VR headset using a powerful rendering computer would provide the performance necessary to implement these additional analysis features.

Finally, additional considerations for improvement on the RESTful Association Grabber tool would search the NCBI Gene database when performing association lookups to determine if returned values are gene or mRNA [19]. The association tool currently performs a general protein to protein association lookup and builds the list of all matches within the original datafile. Addition of a search to NCBI Gene would add the ability to build a list of genes not found in the original datafile that are associate with points in the file during gene expression, which can be visualized in the VR Analysis program.

References

[1] (2017) Low density oligo microarrays. [Online]. Available: https://onlinecourses.science.psu.edu/stat555/node/69

[2] R. Govindarajan, J. Duraiyan, K. Kaliyappan, and M. Palanisamy, "Microarray and its applications," *Journal of Pharmacy and Bioallied Sciences*, no. 4(Suppl 2), p. S310S312, 2012. [Online]. Available: http://doi.org/10.4103/0975-7406.100283

[3] M. K. Kotni, M. Zhao, and D.-Q. Wei, "Gene expression profiles and protein-protein interaction networks in amyotrophic lateral sclerosis patients with c9orf72 mutation," *Orphanet Journal of Rare Diseases*, vol. 11, no. 1, p. 148, Nov 2016. [Online]. Available: https://doi.org/10.1186/s13023-016-0531-y

[4] G. Tzanis, C. Berberidis, and I. Vlahavas, "Biological data mining," *Scientific Programming*, no. 16, 2015. [Online]. Available: https://www.researchgate.net/publication/220060935_Biological_Data_Mining

[5] "What is dna?" [Online]. Available: https://ghr.nlm.nih.gov/primer/basics/dna

[6] C. Pastrello, D. O. K. Fortney, G. Agapito, M. Cannataro, E. Shirdel, and I. Jurisica, "Visual data mining of biological networks: One size does not fit all," *PLoS Comput Biol*, no. 9, 2013. [Online]. Available: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1002833

[7] "A force-directed diagram layout algorithm," 2010. [Online]. Available: http://www.brad-smith.info/blog/archives/129

[8] "How can oculus rift be used for healthcare?" 2014. [Online]. Available: https://www.forbes.com/sites/quora/2014/06/18/how-can-the-oculus-rift-be-used-for-healthcare/#438fc9d86c2b

[9] P. Langfelder and S. Horvath, “Simulation of gene expression data.” [Online]. Available: https://labs.genetics.ucla.edu/horvath/CoexpressionNetwork/EigengeneNetwork/SupplementSimulations.pdf

[10] N. Férey, P. E. Gros, J. Hérisson, and R. Gherbi, “Visual data mining of genomic databases by immersive graph-based exploration,” in *Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, ser. GRAPHITE ’05. New York, NY, USA: ACM, 2005, pp. 143–146. [Online]. Available: http://doi.acm.org.ezproxy.library.ewu.edu/10.1145/1101389.1101418

[11] (2017) Geo datasets. [Online]. Available: https://www.ncbi.nlm.nih.gov/gds

[12] (2005) Anti-leukemia drug etoposide effect on ecotropic viral integration site 1-overexpressing myeloid cells. [Online]. Available: http://www.ncbi.nlm.nih.gov/geo

[13] (2005) Srpin803 small molecule inhibitor of srpk1 effect on retinal pigment epithelial cell line. [Online]. Available: http://www.ncbi.nlm.nih.gov/geo

[14] (2017) Biological general repository for interaction datasets. [Online]. Available: http://thebiogrid.org

[15] (2010) Clustering. [Online]. Available: http://scikit-learn.org/stable/modules/clustering.html

[16] S. Morooka, H. Hoshina, I. Kii, and S. Okabe, “Identification of a dual inhibitor of srpk1 and ck2 that attenuates pathological angiogenesis of macular degeneration in mice,” *Mol Pharmacol*.

[17] W. Torres-Garca, W. Zhang, G. C. Runger, R. H. Johnson, and D. R. Meldrum, “Integrative analysis of transcriptomic and proteomic data of desulfovibrio vulgaris : a non-linear model to predict abundance of undetected proteins,” *Bioinformatics*.

[18] D. Durand and D. Sankoff, “Tests for gene clustering,” in *Proceedings of the Sixth Annual International Conference on Computational Biology*, ser. RECOMB ’02. New York, NY, USA: ACM, 2002, pp. 144–154. [Online]. Available: http://doi.acm.org.ezproxy.library.ewu.edu/10.1145/565196.565214

[19] Ncbi gene. [Online]. Available: https://www.ncbi.nlm.nih.gov/gene

Vita

Author: Joshua M. Cotes

Place of Birth: Spokane, Washington

Undergraduate Schools Attended: Spokane Falls Community College
Whitworth University

Degrees Awarded: Bachelor of Arts, 2008, Whitworth University

Honors and Awards: Phi Theta Capa Honor Society

Professional Experience:

Technical Assistant II, Inland Northwest Blood Center, Spokane, WA 2013 to 2015
Laboratory Assistant, Sacred Heart Blood Bank, 2012 to 2013
Installation Technician, DirecTV Home Services, 2009 to 2011
Laboratory Assistant, Sacred Heart Microbiology Laboratory, 2008 to 2009