Tay Truong

Weiting Liao

Professor Rahul

CSC 664-01

May 22 2023

**Visualization of a Large Number of 3D Points Using Linear Embedding Techniques**

**Introduction**

This project delves into data visualization and exploration techniques of large datasets. High-dimensional data (i.e. data that has more than 3 attributes) is difficult to visualize directly. One must use linear embedding techniques to embed the data in a lower dimensional space, whether it be 2D or 3D, to better support visualization and interactions. As the internet grows more prevalent and integral to our lives, so does the size and dimensionality of the datasets flowing through. Rather than querying from tables, being able to visualize and interact with the data will potentially lead to patterns and connections that may not be so apparent as a bare matrix. The following sections will denote the proposed formulation and what we’re trying to achieve, a high level description of our proposed solution, prior work committed to this area and how our project compares, a description of our work and design, and what this project concludes to. Due to the interactive and visual nature of our project, much of the resulting descriptions and outcomes are highly qualitative.

**Proposed Problem Formulation**

At the heart of our project is the challenge of making high-dimensional data more accessible and easier to understand. To address this, our endeavor focuses on two main objectives:

1.Simplification of High-Dimensional Data: The first step in this process involves the application of linear embedding techniques. These techniques will allow us to transform complex, high-dimensional data into a simpler, lower-dimensional format that can be readily visualized. This involves mathematical methods such as Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) that can retain significant information from the high-dimensional data while mapping it onto a 2D or 3D space.

2.Construction of a 3D Visualization Environment: Once we've transformed the high-dimensional data, our next goal is to build an interactive 3D environment that can represent this data in a visually accessible way. This environment should not only present the data points effectively but also support user navigation and information retrieval. The user should be able to navigate through the sea of points, select individual points, and retrieve detailed information related to those points, thus making the visualization not just a representation, but a tool for exploration and understanding.

Through our project, we aim to not only simplify high-dimensional data but also make it more interactive and revealing. By turning this data into an accessible visual experience, we can help reveal patterns, connections, and insights that would otherwise be difficult to uncover ([3](#kix.dpidcubmiksp)).

Therefore, as you explore our work in the following sections, pay close attention to how we use linear embedding techniques and how we build and utilize our 3D visualization environment to enhance the exploration and understanding of high-dimensional data.

**High Level Description**

Our idea to address the complexities of high-dimensional data visualization hinges upon a blend of data science techniques and 3D visual environments.

Firstly, we plan to implement linear embedding techniques to translate high-dimensional data into a format that can be visualized in 2D or 3D space. Utilizing Principal Component Analysis, a cutting-edge dimensional reduction algorithm, we aim to maintain the essential information encapsulated in the data while rendering it in a form that can be utilized for the visual representation.

Following this transformation, we propose to construct an interactive 3D environment. By leveraging three.js libraries, we aim to generate a visualization that clearly depicts each data point, but also supports user interactions like point selection and navigation.

But we're not just stopping at the 'view'. We're venturing to make this environment responsive, where users can 'click' or 'hover' over points to retrieve more detailed information related to each one. Essentially, we're building an interface that not only represents data visually but serves as a dynamic tool for exploration and understanding.

Our approach makes sense, we offer a solution that blends the scientific rigor of data analysis with the intuitive appeal of visual cognition.

**Prior Work in Area**

Threejs ([4](#kix.l5b7b9uf1o8u)) is a JavaScript library that simplifies the process of creating 3D graphics for the web, and it has a plethora of online tutorials, articles, and forum threads dedicated to it. One of the essential resources is the official Three.js documentation, which offers a comprehensive understanding of the library's features, including how to set up scenes, create geometries, manage lighting, and control cameras. The web-based interactive cubes demo ([5](#kix.ydo5kcjk3sgs)) utilizes it. Three.js and this example provide us with a visible 3D interactive environment and also give us guidance on how to implement some of the fundamentals for our project such as controlling the camera and raycasting. This is a crucial step because the advanced features of our 3D environment are built on this foundation.

The article from Built In ([2](#ccvoby9hjp9g)) provides a step-by-step explanation of PCA, a statistical procedure that orthogonally transforms the original coordinates of a data set into a new set of coordinates, which are the principal components. This has direct relevance to the task of simplifying high-dimensional data, as it allows for the reduction of dimensionality with minimal loss of information.The Built In tutorial, while providing a clear introduction, simplifies many aspects of PCA, and does not explore its limitations or alternatives. Additionally, "The Elements of Statistical Learning" ([1](#kix.d7hvns9nmd6l)) is a significant work in this field. It provides a comprehensive overview of many methods for dimensionality reduction and data simplification, including PCA.These resources inspire our work by highlighting the importance of simplifying high-dimensional data, and by showcasing PCA as a powerful technique for achieving this goal.The textbook, on the other hand, may provide too broad a view, without a detailed discussion of the implementation of each method.

**Project Description**

Data Processing (Tay Truong)

Before we begin constructing the environment, the data must be processed into a usable format. Whilst the metadata can remain untouched, Principal Component Analysis must be applied to the csv files of two datasets: Portugal and Israel, both of which bear over 5000 dimensions of data. We then preserve only 3 components to serve as our coordinates for the environment. Programmatically, both the metadata files (being in a xlsx format) and the csv files must be read and converted into simple arrays for ease of access. The indices of one array corresponds one-to-one to the indices of the other, creating an association between points and metadata. And during this stage, to ensure points do not exactly overlap when generated, a random axis of one of the two exactly overlapping points is shifted positively or negatively.

GUI (Tay Truong, Weiting Liao)

By making use of an external library called dat.gui, we were able to display the metadata of a given point, the number of points that share the same characteristics with the displayed point, and a color key listing notable point types. Additionally, an interactive ‘hotbar’ was created at the top of the display which allows users to input their files and modify certain parameters within the environment. They can toggle between clicking and hovering on points, change the distance multiplier between points, scale points by a multiplier, rotate points about themselves, reset rotations, reset multiplier changes and rotations, and reset the camera back to the origin.

Camera: Orbit Controls (Weiting Liao)

Rather than a static, unmoving camera within our environment, a controllable camera implemented using OrbitControls, an addon for threeJS, allows us to view the data from different angles. Objects within the camera’s frustum are captured and can be viewed. This may allow users to see data points that would otherwise be hidden from the static camera or give us insights and relations in the data we may have not seen before. For ease of use, the ‘pan’ and ‘rotation’ components of OrbitControls are inverted from their default values such that panning is left click whilst rotation is right click.

Camera: Change Point of Rotation (Tay Truong)

By pressing the ‘E’ key whilst a point is selected (as in their respective metadata entry is displayed within the GUI), users are able to change the camera’s point of rotation to be around that selected point. With this, users can hone in on certain groups of points with ease rather than attempting to micro-adjust the camera to a perfect degree.

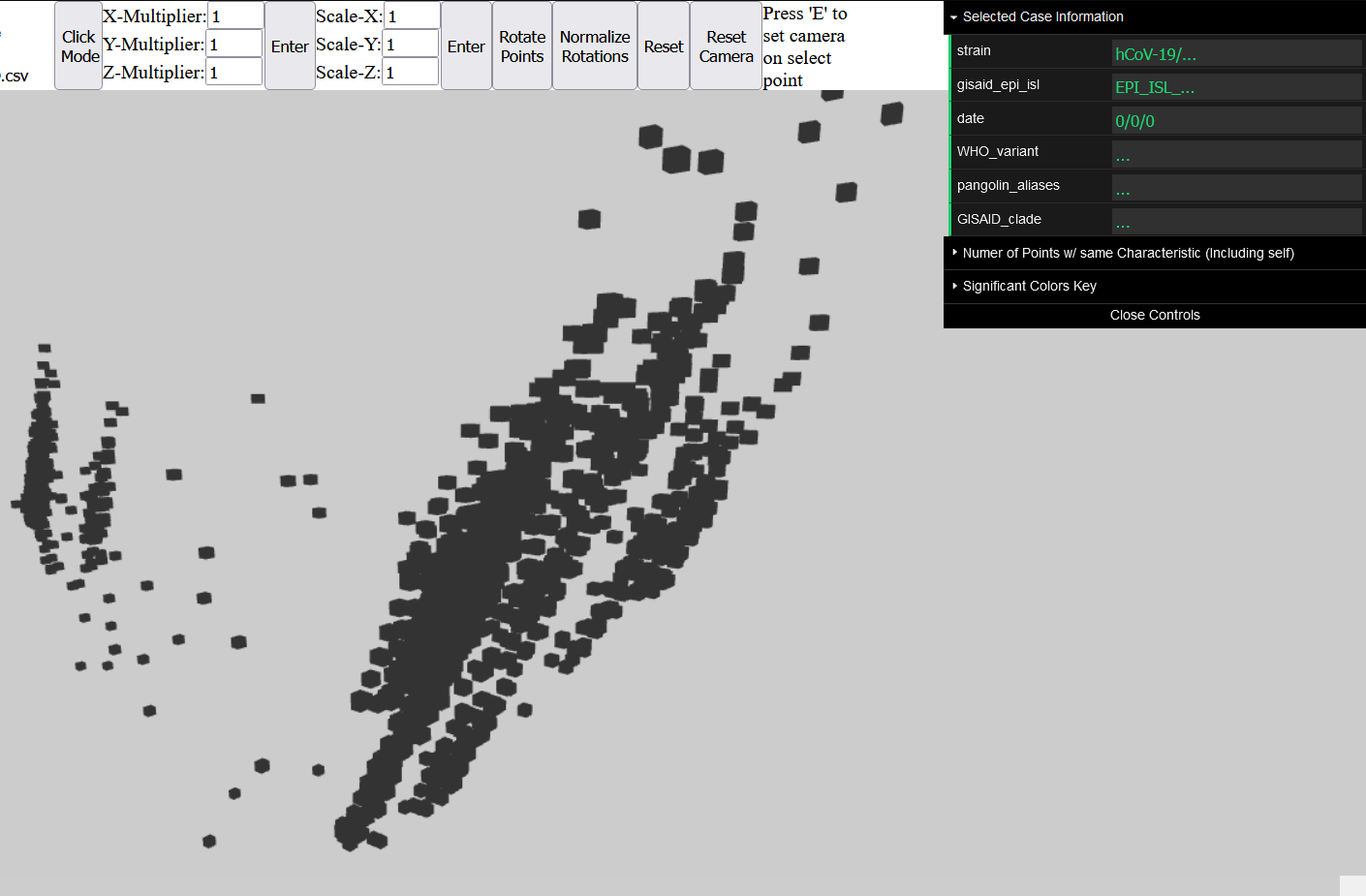
Raycasting & Metadata Retrieval (Tay Truong)

Marking a point in space with the mouse cursor, a ray is cast out from the camera towards the cursor position. If the ray intersects with any objects in the environment, it picks out the first intersected object (as the only objects generated in the space are data points), colors it white so long as the mouse hovers over that point, and fetches metadata about that point. To fetch metadata, the id of a point is stored into an array, from which the index of that id is then queried into the metadata array, thereby fetching the matching entry. To differentiate ‘dragging’ the camera compared to clicking on some point in space, the actions ‘mousedown’ and ‘mouseup’ are distinguished. If the position of the cursor deviates some threshold when transitioning from ‘mousedown’ to ‘mouseup’, a click isn’t registered. Depending upon the toggle state between ‘click’ and ‘hover’ mode, the entries within the GUI are changed.

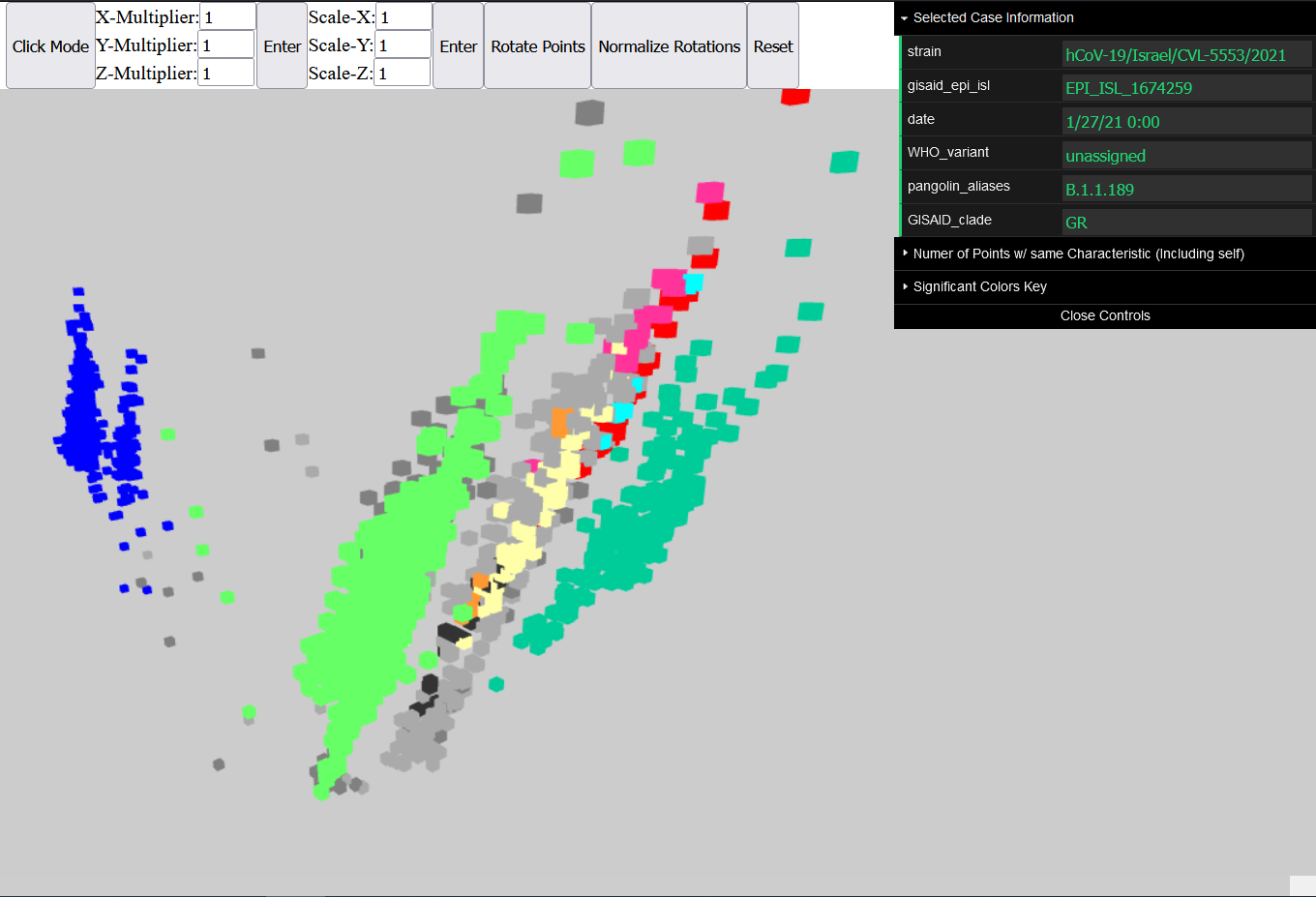
Point Generation & Categorization (Tay Truong, Weiting Liao)

After initialization of the scene and rendering, we traverse through the csv array, generating points and pulling their generated ids into an id array for metadata retrieval. The points are then color coded based on their WHO\_variant; as a majority of points have an unassigned WHO\_variant, we also rely on pangolin\_aliases for color coding. Once the basic properties of that point are defined, we define its coordinates within the environment based on the three components of the csv array as it is finally added to the scene. It should be noted that unlike color and position, the size and rotation of points don’t represent any characteristics as these features are implemented solely for usability.

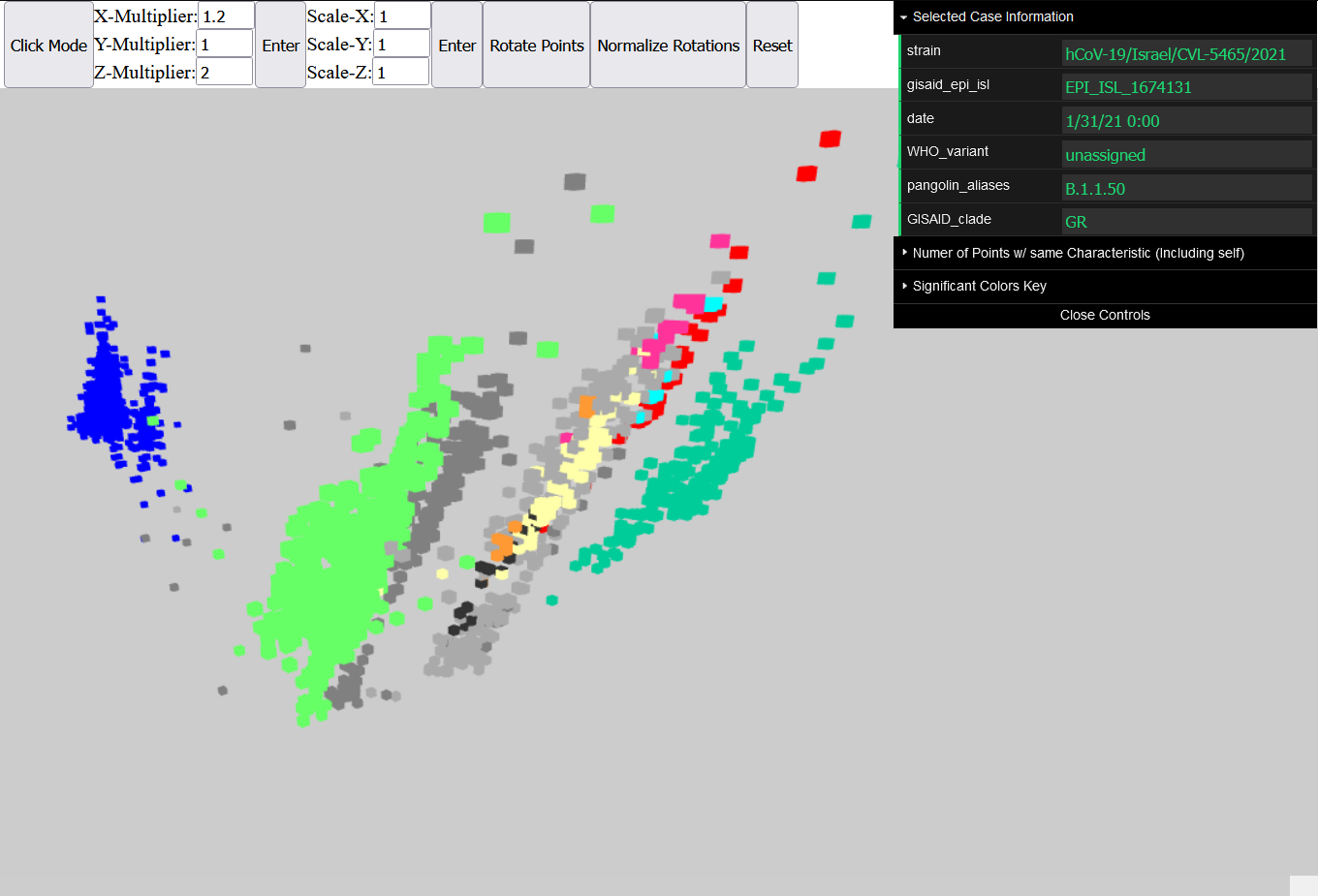
**Experimental Evaluations**

****

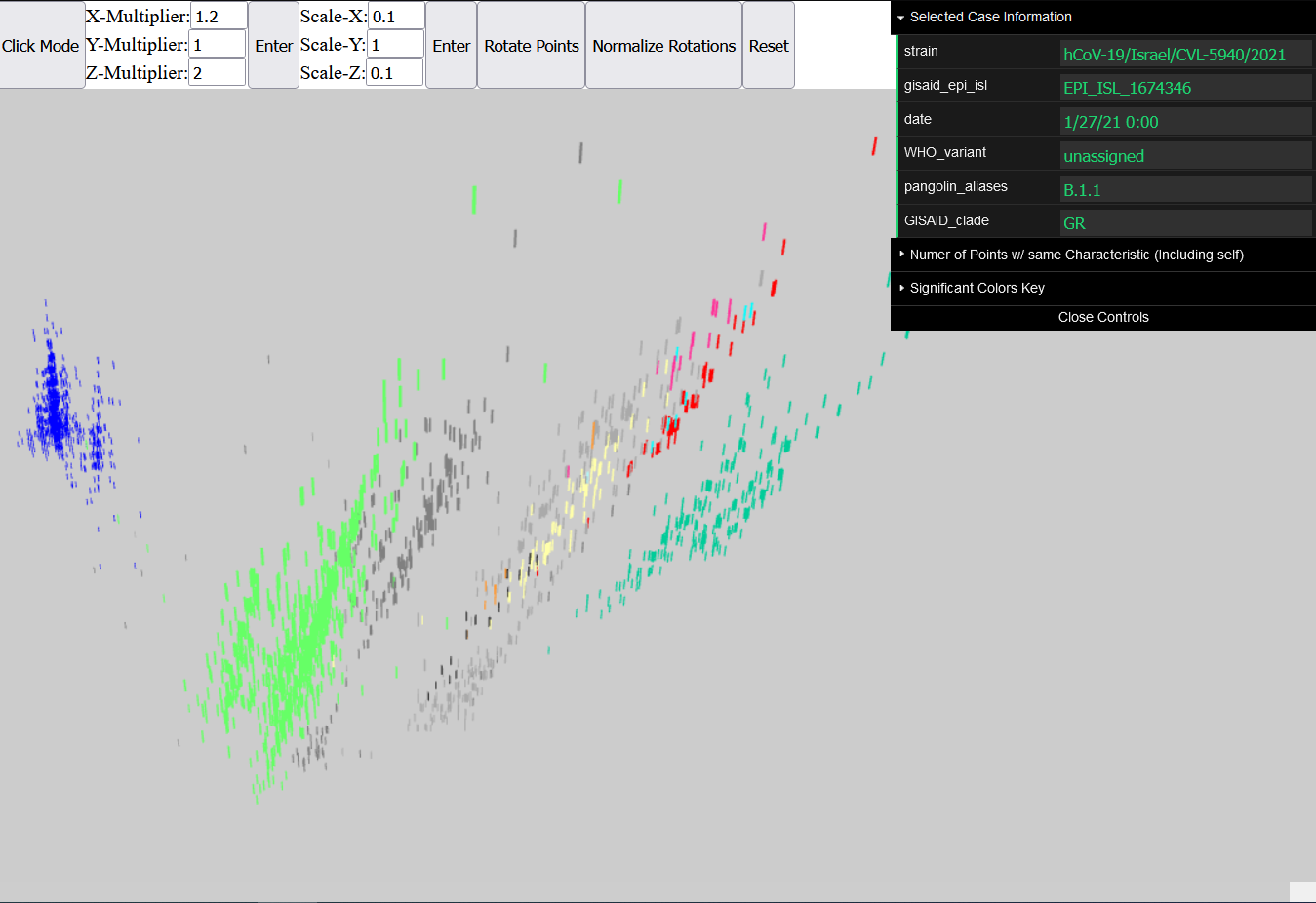
(1a) Israel Dataset - Overview of the 3D Environment without color coding

****

(1b) Israel Dataset - Overview of the 3D Environment at base distance and point scaling



(1c) Israel Dataset - Overview of the 3D Environment at a distance scaling of (1.2, 1, 2)

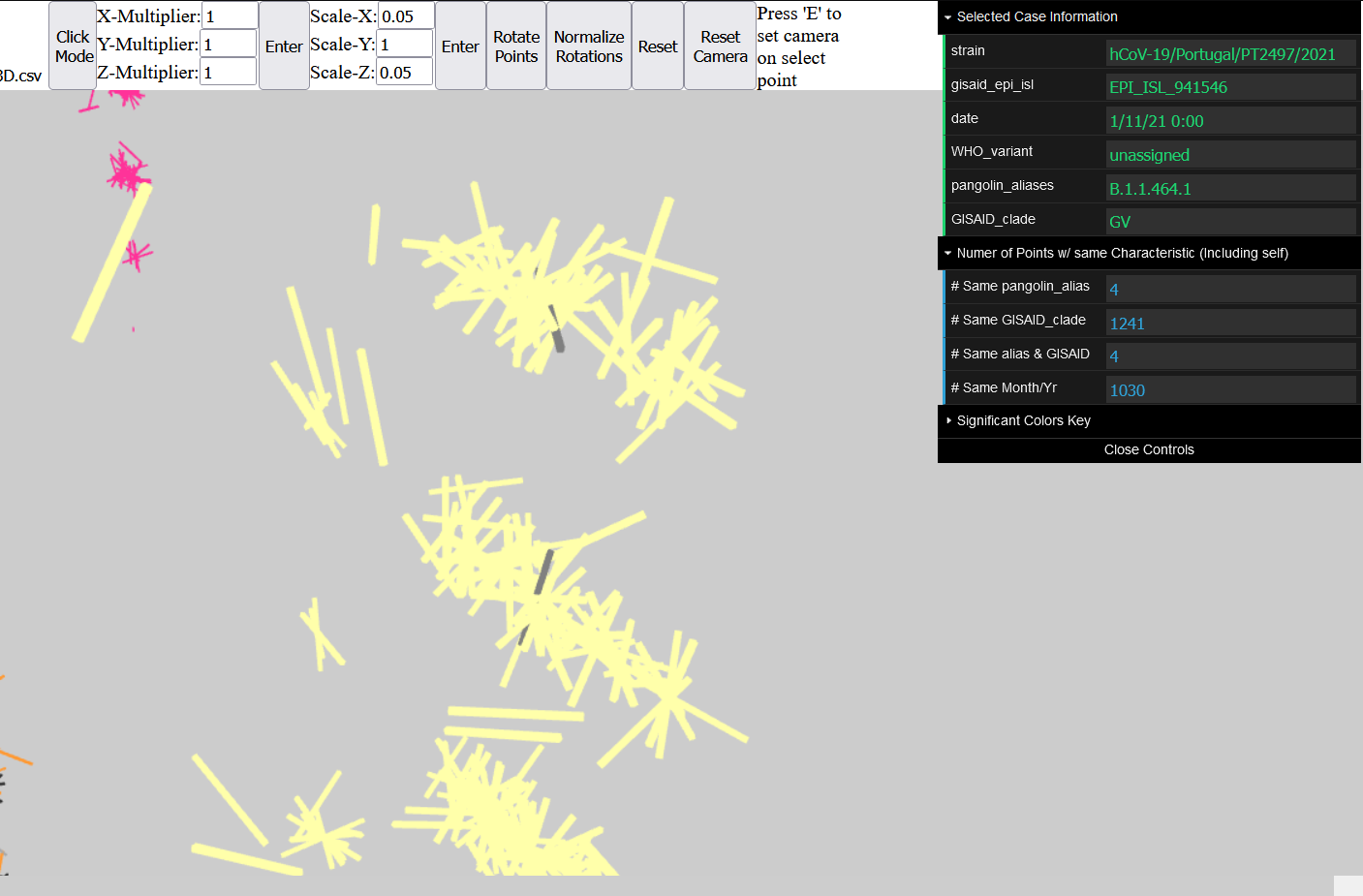


(1d) Israel Dataset - Overview of the 3D Environment at a distance scaling of (1.2, 1, 2) and point scaling of (0.1, 1, 0.1)

The above series of images show the gradual progression of an increase in ‘visualization’ and interactivity as navigating the environment and interacting with the dataset grows less difficult. Image [1a](#87qsyn8wfsa4) shows the environment and its generated points without any color coding. At a glance, a user may potentially distinguish five slanted columns of points and so categorize data points into five groups. Though [1b](#8a0hik9nruxn) denotes otherwise as the columns to the far left are in fact a single defined group of blue points whilst the center column turned out to be a mixture of different point types. Still, there exists large swathes of color wherein individual points within these clusters are difficult to distinguish amongst themselves. Here, one may adjust the distances between all points ([1c](#z6h181t0o4jd)) and/or adjust the size of the points ([1d](#fkd4srdzy1wz)) themselves such that they become more distinct individually. This helps emphasize the large number of points generated within this environment.



(2a) Portugal Dataset - A closeup on point clusters of pangolin\_alias B.1.177.# with the camera’s point of rotation

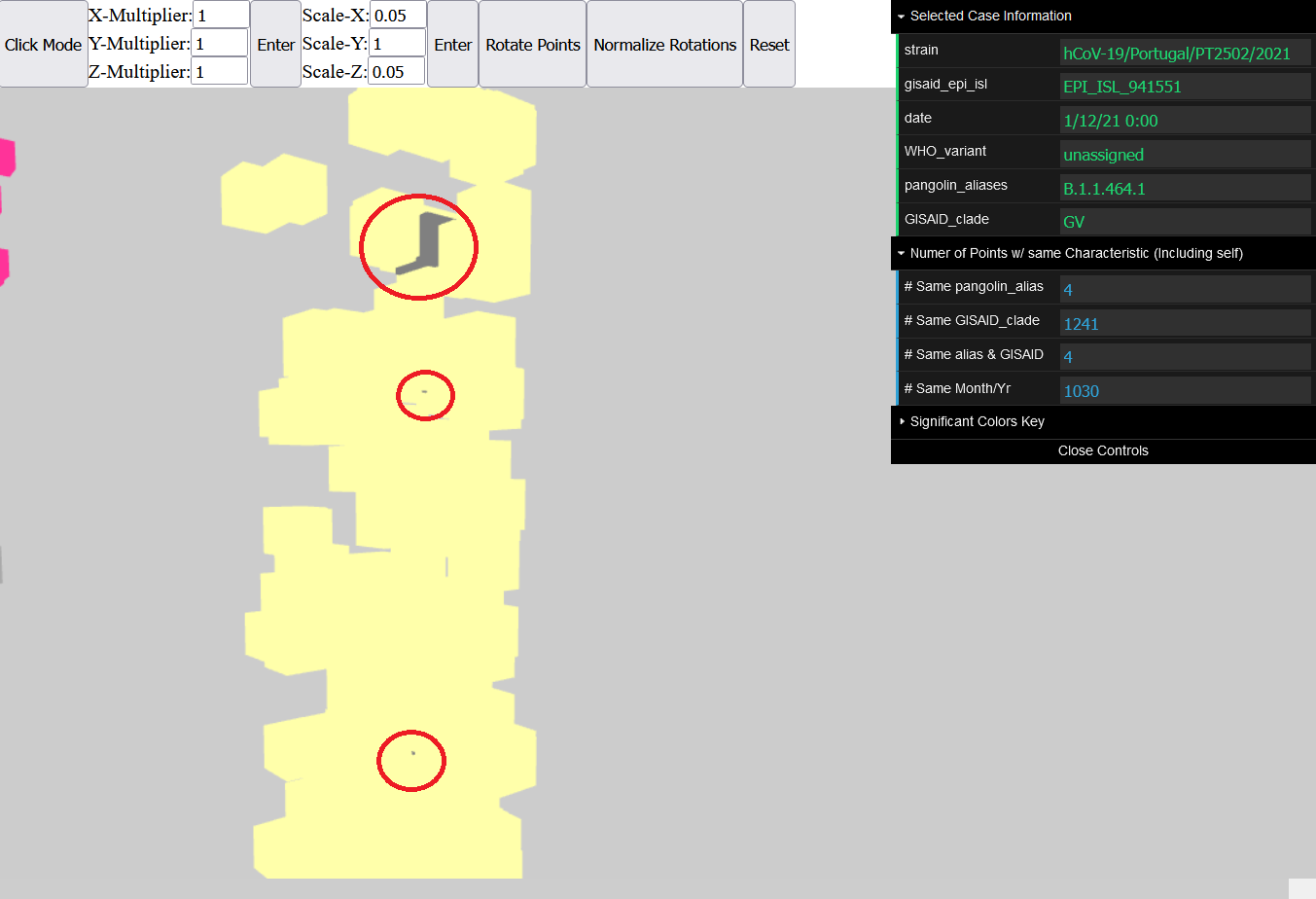


(2b) Portugal Dataset - A closeup on point clusters of pangolin\_alias B.1.177.# with point scaling (0.05, 1, 0.05), point rotation, and with the camera’s point of rotation

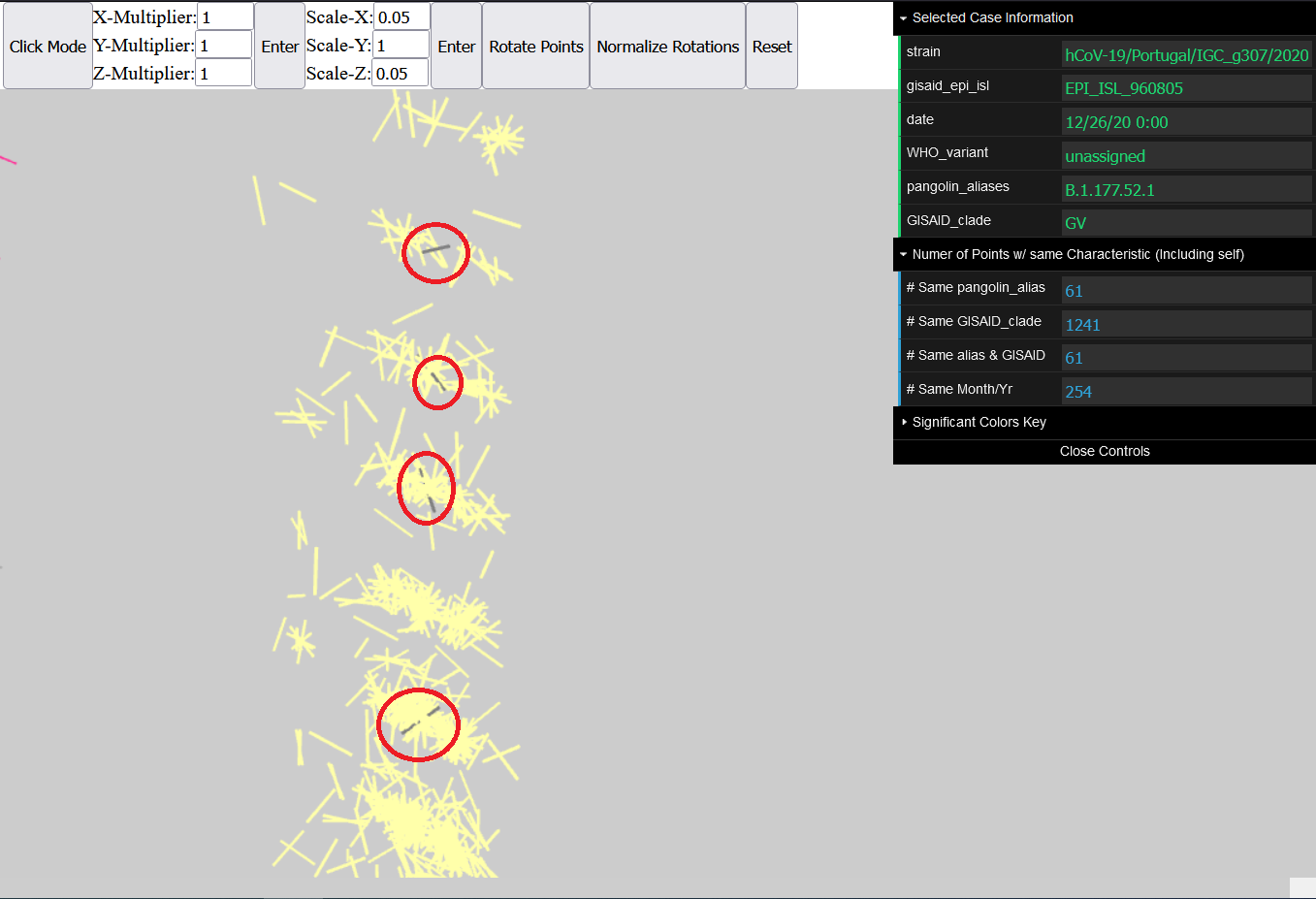
From these closeups, it has become clear that clustering points is an issue of generation and hinders a user's ability to clearly visualize individual data points amongst these clusters ([2a](#xwf19lodd1iy)). The issue is inherent to the dimensional reduction of our datasets as condensing over 5000 dimensions into a mere 3 is bound to be lossy. Not to mention, such components fall into a narrow range of values, each a decimal value and boasting at least ten significant figures; a majority of points are bound to overlap.

This is further emphasized by our implementation of these data points as each point’s relation to every other point in the dataset is defined by their coordinates in this environment. Significant changes to a point’s position while other points remain static would ‘damage’ the information the environment is attempting to convey. Though exactly overlapping points must be changed such that the points could actually be distinguishable, this degree of damage is tolerable as a significant number of exact overlapping points would have adverse effects on a user’s experience within this environment.

On the other hand, majorly overlapping points still remains an issue. We cannot take the same liberties of change as we applied to exact overlaps; changing points within a cluster may cause them to collide and merge with another cluster, destroying that relation. Whilst we may alleviate its symptoms by distance scaling, point scaling, and point rotations ([2b](#rhs79i5ehdg1)), the problem at its core is still relevant.



(3a) Portugal Dataset - A closeup on pangolin\_aliases B.1.177.# where there exists 4 covered gray points of alias B.1.1.464.1. Three are circled whereas the fourth is entirely hidden.



(3b) Portugal Dataset - A closeup on pangolin\_aliases B.1.177.# after applying point scaling of (0.05, 1, 0.05) and a rotation. All 4 instances of gray points are now visible.

Though in the process of addressing clustering points, we’ve given users the ability to uncover points that would otherwise be hidden within the environment. For example, given a rubix cube, one may be able to see and shift all of its outer facets, yet the inner core will remain hidden from prying eyes. Above, we’re presented the same clusters as seen before, though now a few points bearing a different pangolin\_alias are emphasized ([3a](#1puarcx0dye7)). At a glance, one gray point can clearly be seen while another two are barely peeking out; the fourth is entirely buried and hidden at the center of one of the clusters. But by applying distance and scaling transformations as well as a rotation ([3b](#85w50s4i3w1l)), all four points can now be seen and pinpointed within their clusters. Whilst the existence of hidden points in itself is a symptom of our implementation, the ability to reveal such points is still beneficial when examining clusters up close.

**Conclusions**

This project focuses on exploring and visualizing high-dimensional datasets to make them more comprehensible and accessible. The objectives were to simplify high-dimensional data through linear embedding techniques, to construct an interactive 3D visualization environment, and to potentially give users new insights to the dataset they wouldn’t have found any other way.

Regarding the first two goals, our project was successful in achieving these objectives. High-dimensional data was transformed into a visualizable lower-dimensional format, and an interactive 3D environment was built to represent this data visually. While we could have gone further in emphasizing points by adding edges, that was unfortunately unable to be achieved as it created an incompatibility with our raycasting. The environment is responsive, allowing users to click or hover over points to retrieve more detailed information. As for whether our implementation evokes gaining insight rather than simple information retrieval, to a degree, we believe this has been somewhat accomplished as data points have been grouped by pangolin\_aliases. Though, work can be done to improve the ‘insight’ gained through our environment.

The project took inspiration and guidance from resources like the Three.js documentation and Built In’s tutorial on PCA. These resources helped us set up the scene, create geometries, manage lighting, control cameras, and apply PCA for data simplification. Thus, the project successfully delivered its original goals of simplifying and visualizing high-dimensional data.

Going further, we may consider attempts at ‘untangling’ clusters in a manner that doesn’t distort the information that these points convey. As we examine point clusters with closer scrutiny, we also should consider modifying the frustum and camera controls to allow minor adjustments and to limit the capture of points. Additionally, steps can be taken to emphasize and distinguish individual points by manipulating more aspects about them such as adding transparency, texture, mesh shape, and edges. These characteristics will not only help distinguish points, but also allow us to represent additional dimensionalities of the data, representing more information and potentially creating more relations and patterns. The project has taken its first steps, yet there’s still so much more that can be done.

**References**

1. Hastie, T., Tibshirani, R.,, Friedman, J. (2001). The Elements of Statistical Learning. New York, NY, USA: Springer New York Inc.

1. Jaadi, Zakaria. “A Step-by-Step Explanation of Principal Component Analysis (PCA).” Edited by Brennan Whitfield and Sadrach Pierre, *Built In*, 29 Mar. 2023, [builtin.com/data-science/step-step-explanation-principal-component-analysis](https://builtin.com/data-science/step-step-explanation-principal-component-analysis)

1. R. Singh and R. Jain, “From Information-Centric to Experiential Environments”, in Interactive Computation: The New Paradigm, D. Goldin, S. Smolka, and P. Wegner, eds., Springer Verlag, pp. 323 – 351, 2006

1. *Three.Js – JavaScript 3D Library*, [threejs.org/](https://threejs.org/)

1. “Three.Js.” *Three.Js Examples*, [threejs.org/examples/#webgl\_interactive\_cubes](https://threejs.org/examples/webgl_interactive_cubes.html)