Information Visualization: Milestone 3

Project: *Console Wars*

Team: *Marks & Channels*

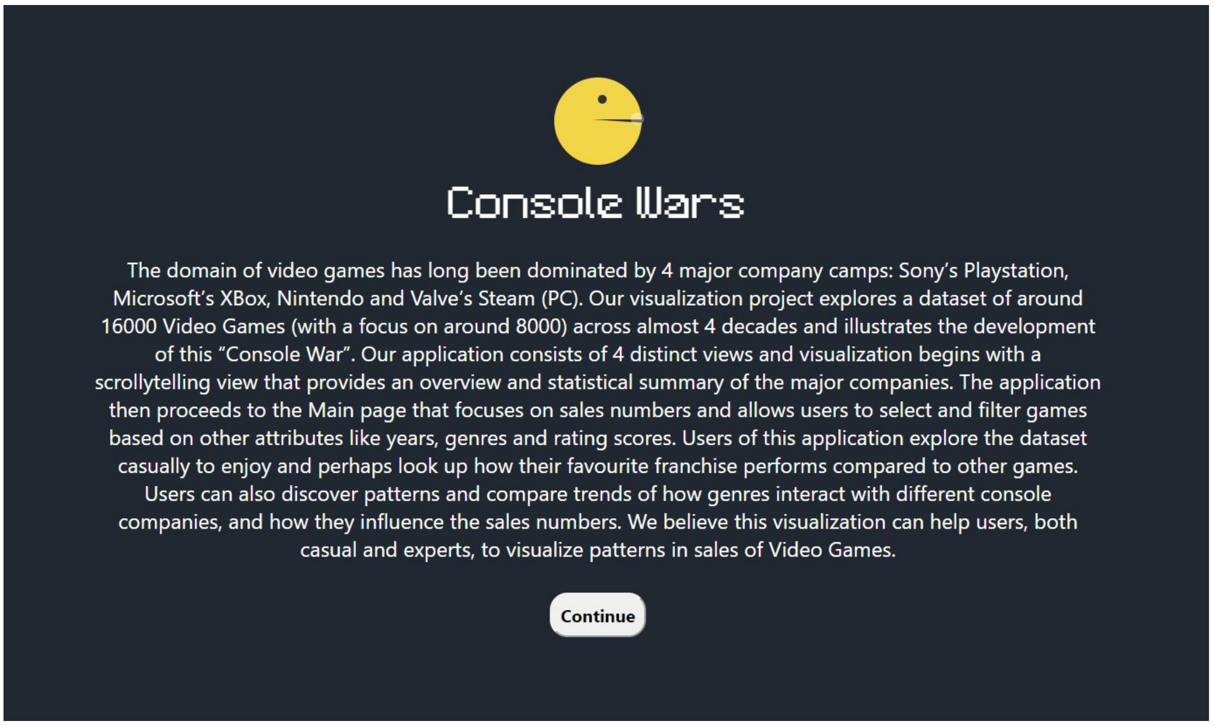
April 10, 2020

v1.0-final

1-Overview

Teaser Image

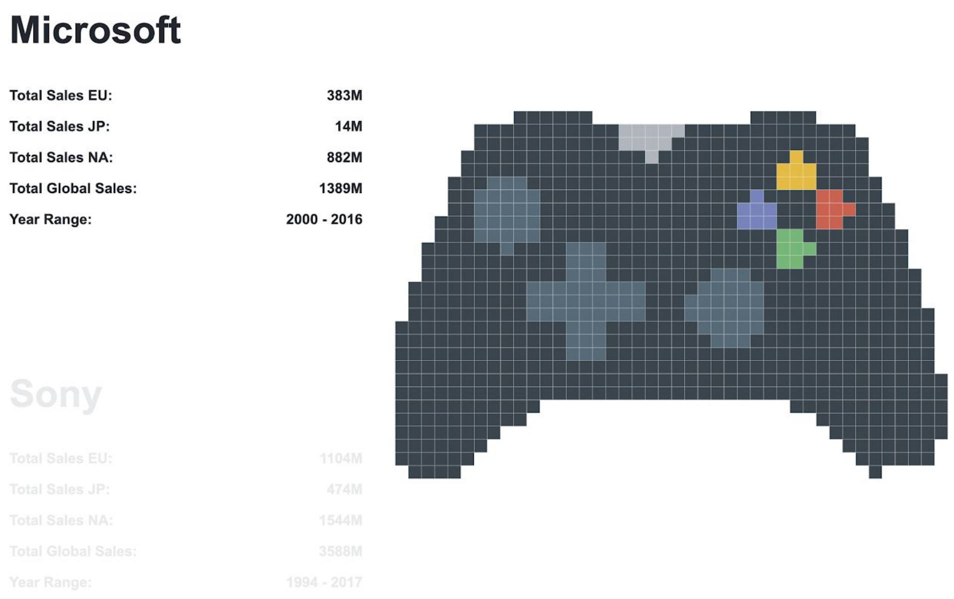
Main page:



Console Wars

The domain of video games has long been dominated by 4 major company camps: Sony's Playstation, Microsoft's XBox, Nintendo and Valve's Steam (PC). Our visualization project explores a dataset of around 16000 Video Games (with a focus on around 8000) across almost 4 decades and illustrates the development of this "Console War". Our application consists of 4 distinct views and visualization begins with a scrollytelling view that provides an overview and statistical summary of the major companies. The application then proceeds to the Main page that focuses on sales numbers and allows users to select and filter games based on other attributes like years, genres and rating scores. Users of this application explore the dataset casually to enjoy and perhaps look up how their favourite franchise performs compared to other games. Users can also discover patterns and compare trends of how genres interact with different console companies, and how they influence the sales numbers. We believe this visualization can help users, both casual and experts, to visualize patterns in sales of Video Games.

Controller View (Scrollytelling):



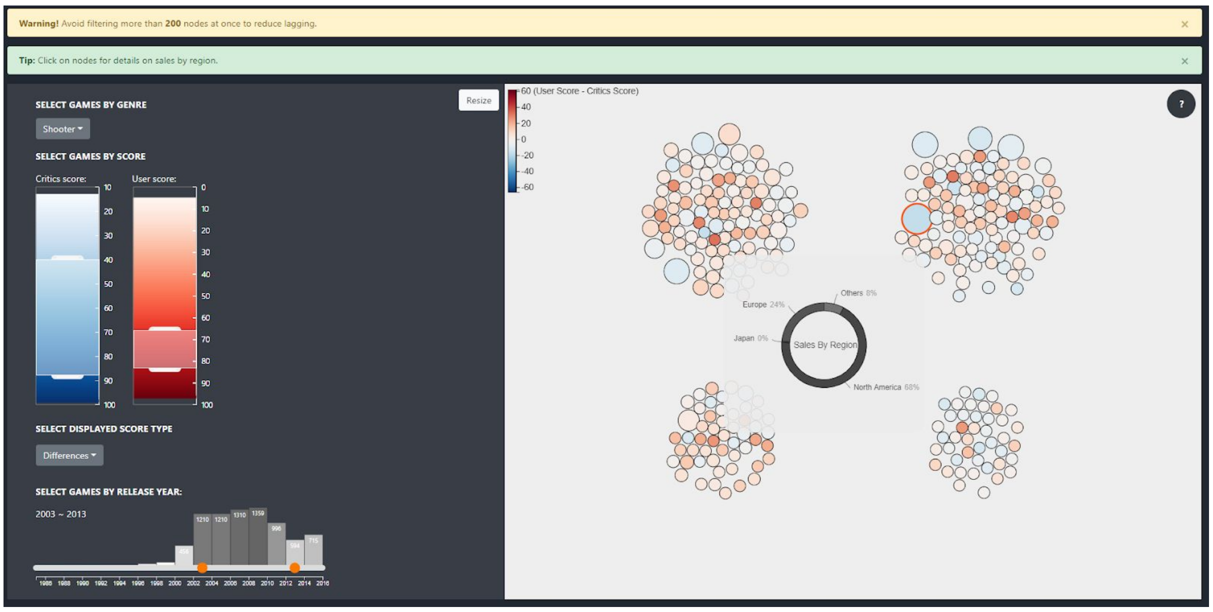
Microsoft

Total Sales EU:	383M
Total Sales JP:	14M
Total Sales NA:	882M
Total Global Sales:	1389M
Year Range:	2000 - 2016

Sony

Total Sales EU:	1104M
Total Sales JP:	474M
Total Sales NA:	1544M
Total Global Sales:	3588M
Year Range:	1994 - 2017

Main view, Widget and Sales-by-region view:


Warning! Avoid filtering more than **200** nodes at once to reduce lagging.

Tip: Click on nodes for details on sales by region.

Summary of Project

The domain of video games has long been dominated by 4 major company camps: Sony's Playstation, Microsoft's XBox, Nintendo and Valve's Steam (PC). Our visualization project explores a dataset of around 16000 Video Games (with a focus on around 8000) across almost 4 decades and illustrates the development of this "Console War". Our application consists of 4 distinct views and visualization begins with a scrollytelling view that provides an overview and statistical summary of the major companies. The application then proceeds to the Main page that focuses on sales numbers and allows users to select and filter games based on other attributes like years, genres and rating scores. Users of this application explore the dataset casually to enjoy and perhaps look up how their favourite franchise performs compared to other games. Users can also discover patterns and compare trends of how genres interact with different console companies, and how they influence the sales numbers. We believe this visualization can help users, both casual users and experts, to visualize patterns in sales of Video Games.

2-Data

Description of Dataset

In domain-specific terms, our dataset describes the sales performance, users/critics rating and basic information for video games released between 1985-2016. The user can view summary statistics about the dataset based on the type of platform.

In abstract language, our data source is of type **table**, consisting of 15 attributes and 16718 items. After going through the preprocessing pipeline, the dataset that we visualize consists of 14 attributes (14 visualized but some more for programming shortcut) and 8085 items:

- ***Name*** is the name of the game. It is a **categorical** attribute and has cardinality of **5052** (Games can have same name, but released on different console)
- ***Platform*** is the console that the game released on. It is a **categorical** attribute, and has cardinality of **16**
- ***Genre*** is the genre of the game. It is a **categorical** attribute and has cardinality of **12**
- ***Publisher*** is the publisher of the game. It is a **categorical** attribute and has cardinality of **303**
- ***Year*** is the year the game is released on. It is a **quantitative** attribute since some operation e.g. differences can be meaningful, and has range of **[1985 - 2016]**
- ***NA Sales*** is the number of copies (in Million) of the game sold in North America. It is a **quantitative** attribute and has range of **[0 - 41.36]**
- ***EU Sales*** is the number of copies (in Million) of the game sold in Europe. It is a **quantitative** attribute and has range of **[0 - 28.96]**
- ***JP Sales*** is the number of copies (in Million) of the game sold in Japan. It is a **quantitative** attribute and has range of **[0 - 6.5]**
- ***Others Sales*** is the number of copies (in Million) of the game sold in other regions. It is a **quantitative** attribute and has range of **[0 - 10.57]**
- ***Global Sales*** is the number of copies (in Million) of the game sold in the world. It is the sum of all the sales attributes above. It is a **quantitative** attribute and has range of **[0 - 82.53]**
- ***Critics Score*** is the average critics rating of the game. It is a **quantitative** attribute and has range of **[13, 98]**
- ***Users Score*** is the average users rating of the game. It is a **quantitative** attribute and has range of **[5, 96]**
- ***Score Difference*** is a derived attribute, which is equal to (Users score - Critics score). It is a **quantitative** attribute and has range of **[-66, 62]**
- ***Console Company*** is a derived attribute, which is the company that owns the console the game is released on. It is a **categorical** attribute and has cardinality of **5** (“Sony”, “Microsoft”, “Nintendo”, “PC”, “Others”)

URL to Dataset

https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings

Preprocessing Pipeline

We modified the original dataset first locally in Excel by removing items that did not have a game name, which resulted in removal of 2 items. The resulting file is the CSV we used in this project. We noted some items have “N/A” for some of its attributes like Year. We removed games with the “N/A” for Year because it cannot be sorted by widgets in the Widget View and there are too many to fill in manually. This resulted in removal of 269 games.

There were some parsing errors in the original dataset and some values for Genre were replaced with the value for Platform. To mitigate these errors, we modified the values for Genre from Platform to “Others”, allowing us to group these games.

In terms of runtime processing for Main View and Widget View, we are parsing the dataset in **preprocessor.js** (Note: for the Controller View, this was done in **controller-data.js**). Since our visualization also shows Users and Critics ratings, we filter out all the items that do not have a Users and Critics rating, pruning away around 8K items and leaving the filtered dataset at around 8K items. 14 games not released on the 4 major conole camps are also pruned, as the small number of items caused the cluster to be empty most of the time. We also created a derived attribute for the difference in score between users and critics rating, for users of the visualization to see how they differ in different consoles or genres. Furthermore, since some games are released on multiple consoles (under the same name), we also linked these same games by including an attribute of an array that stores the ids of the games with the same name (that are released in other consoles). Using this derived attribute, we are able to connect these items in the table to simulate the characteristics of a network. The filtered data will be formatted and set in their corresponding view Class. (i.e. `mainView.data = formatDataForMainView()`). The filtered dataset then gets further divided into 4 dataset arrays by their console company.

The preprocessing steps for the Controller View included finding suitable SVGs for each of the 4 console companies (5 including Others). Once those SVGs were found, we used Adobe Illustrator to create new paths for each of the svgs. These paths ranged from the outline of the svg (in order to determine the area) to tiny details about the controllers. Each path was then converted to x,y coordinates. Using all the path information, we calculated the exact x,y coordinates for each rect on the screen. Different colors were also given to each of the rects on the screen based on the type of controller. For each console company, we saved a file with the fields: id (unique id), x,y coordinates, color, side (length of the side of the rect in that controller). The code for the preprocessing of each of the controller’s data points is in the code base.

In an effort to reduce the number of points being displayed on the screen, any game shared by more than one console company only appears in one of the controllers. The distribution of points follows closely the original distribution and is hence representative of the original dataset. Since the sales data should not be affected by any other fields, we did not remove any data points while calculating these values.

3-Goal and Tasks

In domain-specific terms, our visualization is largely intended for casual game enthusiasts. The goal of the visualization is for the audience to enjoy **browsing different games** and potentially look up data for their favourite game. For example, users can look up Sports games released between 2005 and 2010 and locate the game “Wii Sports” as an **outlier** in sales. Users interested in the history of video games can **explore the development of “Console War” (mostly in terms of sales numbers)**. For instance, users can select a genre and increment the range of years to see how the clusters form patterns or develop. In terms of non-causal analysis, our visualization may also be able to support basic tasks of **game market analysis**, for example determining whether there is a trend for

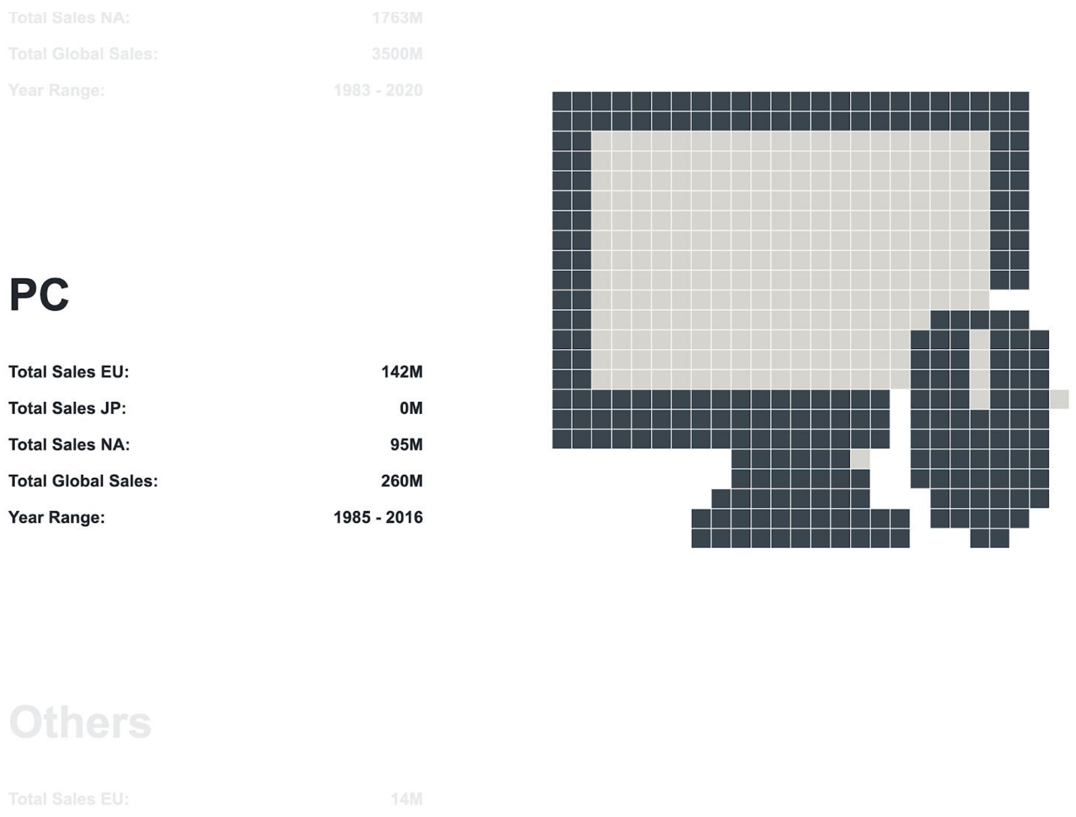
Shooter games to perform better if released on a particular console, in terms of sales performance and rating scores. (comparatively, Shooter games seem to not do too well if released on Nintendo). Also, market analysts may also be able to tell if a genre of games is saturated or dominated by a particular console by comparing cluster sizes, thus informing the user's market decisions. The visualization also allows game enthusiasts to view sales statistics for different console companies. The user can scroll through the various statistics. Viewing the various controllers, the user can gauge the relative number of games released for each of the consoles over a year range.

Our visualization is intended to support the main tasks of **browsing** and **exploring** the dataset **casual**. In terms of non-casual analysis, the visualization also supports **discovering patterns** through filtering and **identifying trends** across a range of years. It also offers **annotation** through tooltips. Further, our visualization also allows users to **locate outliers**. The Controller View **summarizes** the dataset by platform companies and serves as a starting point for the visualization.

4-Visualization

Our visualization project consists of 4 views in total: the Controller View, the Main View, the Widget View and the Sales-by-region View. The Controller view is a summary overview that allows the user to look at summary statistics and data points in the form of different game controllers. The users are able to view summarised information about each platform. The Main View allows users to view the games in their corresponding clusters (grouped by game platform) and to interact with individual nodes to view more information. The Widget View allows users to view an overview of the games meeting filter criteria with the widget pane bar chart, as well as apply different filters on different attributes of a game. We did not implement linked highlighting between the Main View and the Widget View as users cannot see the highlighted node in the Widget View. To compensate for this, the Widget View (discussed below) allows users to filter items within that View and the Main View is updated accordingly. The Sales-by-region View allows users to view the part-whole relation of sales. This view is accessible by selecting nodes in the Main View.

<u>Controller view</u>:



Total Sales NA:	1763M
Total Global Sales:	3500M
Year Range:	1983 - 2020

PC

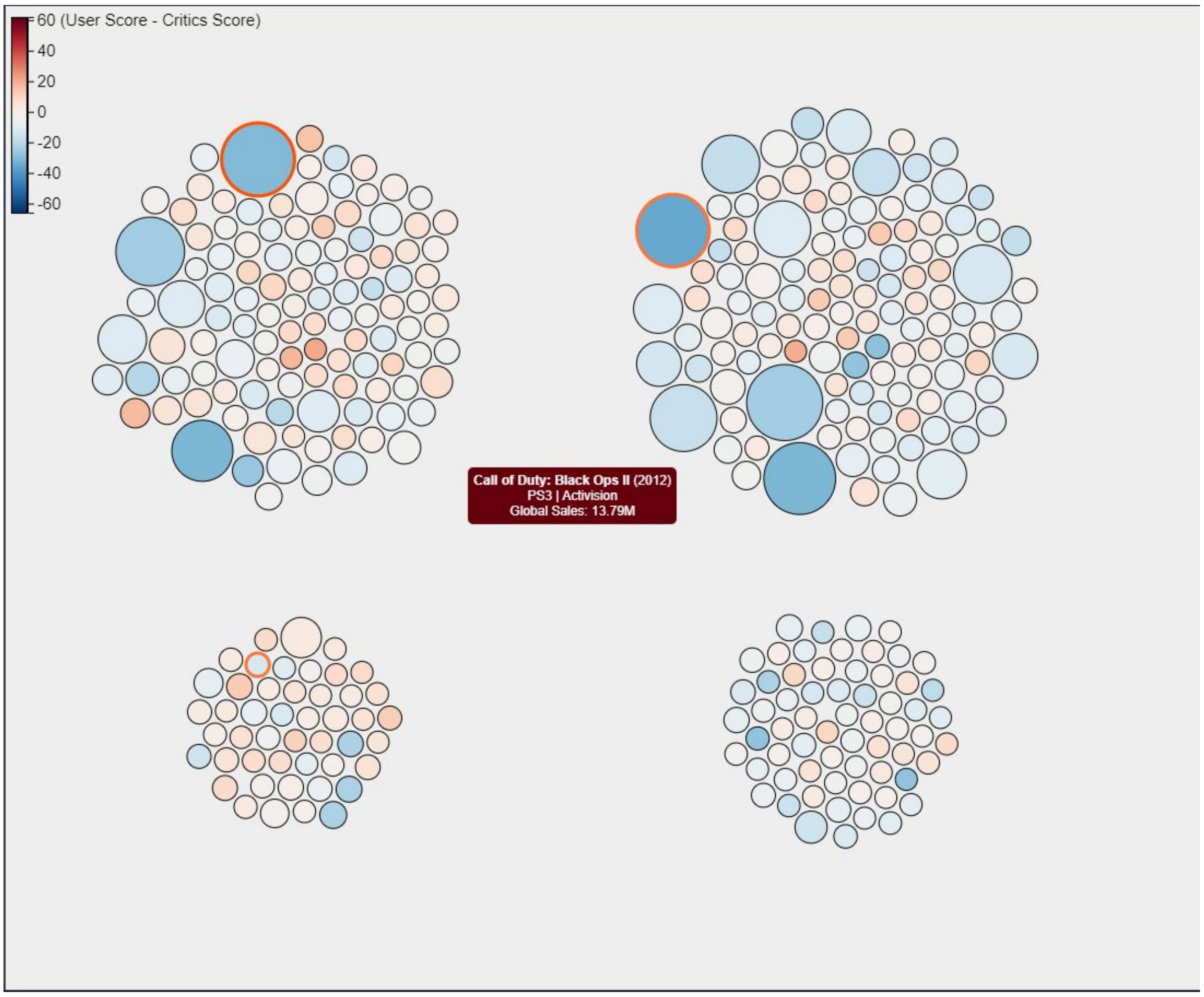
Total Sales EU:	142M
Total Sales JP:	0M
Total Sales NA:	95M
Total Global Sales:	260M
Year Range:	1985 - 2016

Others

Total Sales EU:	14M

In the controller view, each unique game is represented as a **point mark**. The point marks are represented as 5 controllers. Each of the controllers was chosen to represent the major console companies. The **size of each point** is inversely proportional to the number of games for that console company. The users are able to scroll through the multiple statistics while viewing a visual representation of the console company. The view serves as an **overview of the sales data** before the user is able to navigate to look into the dataset in more detail.

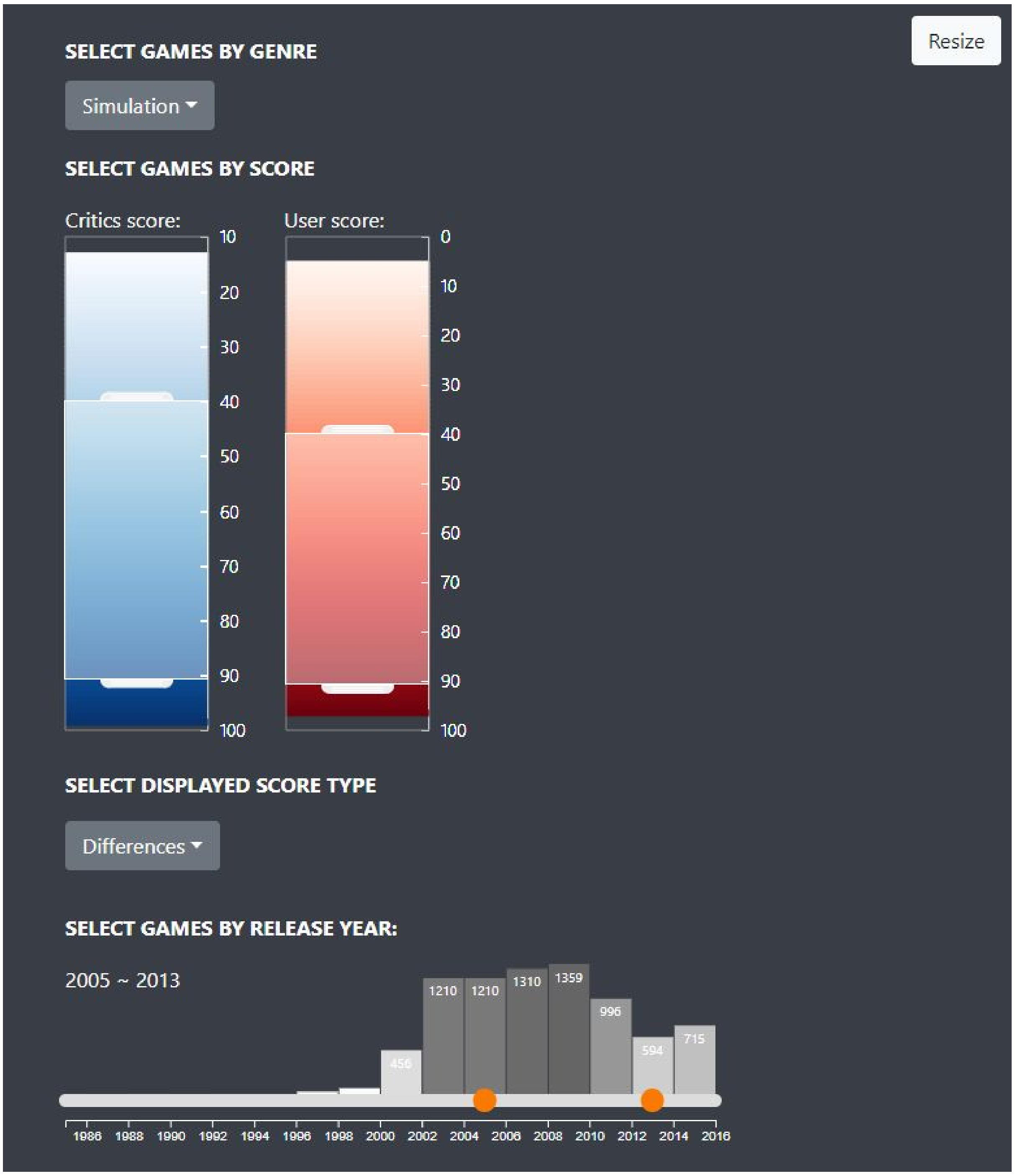
Main view:


60 (User Score - Critics Score)
40
20
0
-20
-40
-60
Call of Duty: Black Ops II (2012)
PS3 | Activision
Global Sales: 13.79M

In the Main View, each game is represented by a **point** mark (circle node). In terms of visual encoding, the **Platform Company** is encoded by **spatial region** with the 4 clusters (Sony in the top left; Microsoft in the top right; Nintendo in the bottom left; PC in the bottom right). Since most tasks involve comparing across companies, we chose to use spatial region, which is the most effective identity channel. Note that the spatial position of nodes within each cluster does not encode anything, and is determined by force simulation. The **Global Sales** number is encoded by **2D size** of the node. While there are more effective magnitude channels, area is the most effective for a point mark (after position but force simulation discarded that above) Different color hues are used to encode the 3 rating-related attributes, and users can use the dropdown in the Widget View to select which rating-attribute to currently encode. **Users Score** is encoded by color **luminance** (Red) and Critics Score is also encoded by color **luminance** (Blue). The **Score Difference** is encoded by both color **hues** (Red and Blue) and **luminance** using a diverging color scale with white denoting 0. (More red represents users rated higher than critics; and more blue represents critics rated higher than users) If users choose to display the score difference, a legend for the diverging color scale will be displayed in the top left corner of the Main View. The divergent color scale allows users to spot any patterns of score difference in different clusters or genres. On hover, the game will be selected, along with games in other clusters with the same name and a tooltip will show the name, year, platform, genre and global sales of the game as supporting figures. On click, the Sales-by-region View for that game will be shown.

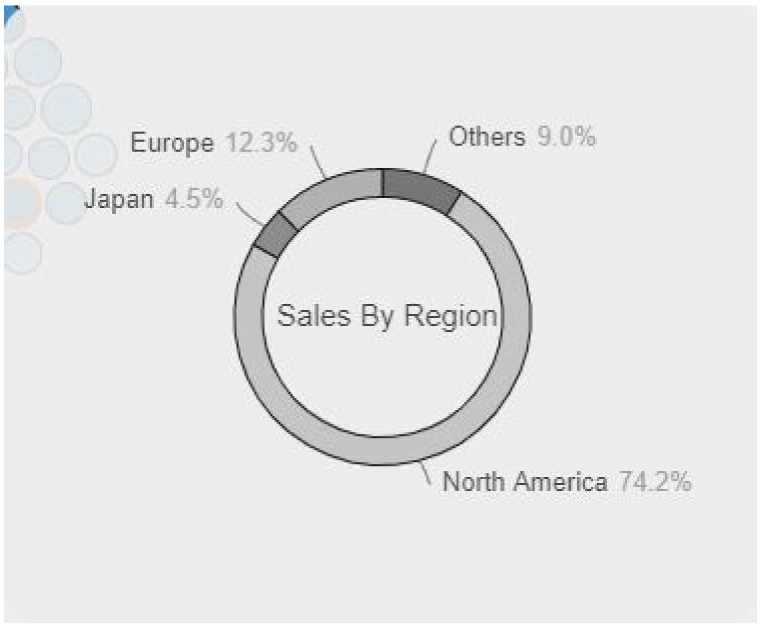
The Main View is linked with the Widget View, as changes in widgets and dropdowns will filter the data in the Main View. The Main View is also linked with the Sales-by-region View by clicking on a node as mentioned above.

Widget view



The widget pane includes a view (**scented widget** for years of game release) and widgets (score slider and dropdowns). The range slider for game scores and dropdown to select different game genres were chosen for their clear and straightforward use. With the scented widget, users can see how the games are distributed by year, and then choose a range they are interested in. The **horizontal position** encodes the **years**, and **vertical position** encodes the **number of games** (released in that 2 year range, or in a single year). **Color Luminance** is also used to encode the number of games, and shades of grey is chosen to match the color scheme of the webpage. The critics and users score controller can also act as **legends** for the Main View, as it shows that higher **saturation** denotes higher rating score, which is self-documenting. The Widget View is linked to the Main View and the changes in widget pane (and other filtering options) will determine what will be shown in the Main View. For example, reducing the default range of user scores to between 80 and 90 can let visitors see what video games earned higher satisfaction from the users who played the game by only showing the game nodes with user scores that fall in the range, on top of other filters that are already applied. This is needed since it allows users to control how much of the data and the specifics of the data to be displayed and can select ranges according to their interests and tasks.

Sales-by-region View



In the Sales-by-region View, a donut chart is used to display the sales numbers of different regions. We used donut chart over pie chart as it can visualize with the same effectiveness but is able to save screen space by allowing the title to be placed in the middle. **Line** marks are used here, as it only allows changes in 1D (length). The **percentage of regional sales** (over global sales) is encoded by **length** (and **curvature**). Different shades of grey are used to match the color of the webpage. This view is only accessible from the Main View by clicking on a node and users can click to toggle between tooltips and Sales-by-region View, since it can save screen space and only overlay it over the Main View if users want to view the information.

5-Reflection

Originally, our visualization plan was not communicated well in our initial proposal. After talking with the TAs, we then finalized on 3 views (Main, Widget and Controller). However, for Milestone 2, the widget view was not fully implemented and the controller view was in another branch. After receiving feedback from TAs, we worked harder to fully implement and integrate all 3 views and added 1 more view (Sales-by-region) for the final product.

While the original proposal did not communicate the idea and plan well, our original proposal was mostly realistic in terms of what is actually technically possible in d3. For the Controller View, the hardest part of finding or creating the perfect paths to encompass all the points while making the outcome look like a controller. Though we weren't sure if this was even possible, we tried creating paths in external programs and ultimately concluded that it could be done. Thus, we did not modify our visualization goal much, but we added more to our technical goal with the use of external programs. We originally wanted to create tooltips for the controller view to allow the user to explore the dataset. However, because of the number of points being displayed and their varied widths, the tooltips do not work as well as we expected them to. For the Main View, however, we originally wanted to make the clusters be shaped into the logos of the company with `d3.force` as animation. However, this proved to be too complicated, so we decided to give up the idea of the logo shapes and focused on `d3.force` (forming the clusters into a circle)

We realized too late into the development process that `d3.force` does not work well for our purposes as the number of nodes increases to above ~200. It also causes some instability in force simulation and sometimes nodes can wander off or the clusters are centered around some other points. We tried to take on feedback from the TAs to maybe replace the force simulation with matrix or swirly layout, but it proved to be too much change to the codebase with too little time left. If we were to start over from scratch, we would have communicated our plans in more detail, and avoid using `d3.force` as our dataset is rather large. Furthermore, using static layouts like matrix or swirls instead of dynamic force simulation allows us to control the alignment better and allow for more quality of life improvements, such as adding logo labels to the clusters. Additionally, it would have been useful to look into the dataset with more detail and devising new and more creative ways of reducing the dataset size. To better illustrate the development of the "Console War", we could have provided other overview views like line charts to communicate trends more directly.

6-Team Assessment

Megha worked on the **Controller View**. Tasks included:

1. Finding/Creating feasible svgs for each controller type.
2. Dividing paths from the each of the svgs in terms of controller outline, controller details
3. Converting all paths to x,y coordinates
4. Rendering the points inside the paths
5. Implement scrollytelling for the view.
6. Process data to derive summaries.
7. Work on some parts of the writeup.
8. Title and Description Page

Wesley worked on the **Main View** and the **Sales-by-region View**. Tasks included:

1. Preprocess the original dataset by filtering and deriving new attributes
2. Create circle nodes and position them in clusters with force simulation
3. Implement node selection, linked highlighting between clusters (*within* view)
4. Create Interactive Tooltips on hover
5. Create color scales and legends based on selection in Widget View
6. Added score difference option for dropdown in Widget View and added diverging color scale
7. Create Donut Chart (Sales-by-region View) based on User's selection in Main View
8. Work on most parts of the writeup

Nicole worked on the **Widget View.** Tasks included:

1. Filtering logic: designing and implementing how data clusters in Main View will be filtered by components in Widget View and how the filtered data will be communicated between the two views.
2. Create 3 widgets and a scented widget to filter rows by columns and define initial filter values to improve rendering of nodes:
    a. dropdown: `genre`
    b. range slider: `user_score` and `critic_score`
    c. dropdown for selecting different game score types (critic, user, or difference between critic and user scores) to be encoded as color in the Main View
    d. scented widget: select range of `year` of game releases using a slider under histogram of number of games, and display current selected range over the histogram
3. Show interactive messages to inform and guide user
4. Overall CSS of the visualization, design layout and alignment of views to work well across the page, including making the Main View more responsive to current/change in user screen size so that they continue to align properly.
5. Project restructuring and code refactoring, including refactoring setup of views and the Widget View class, and using separate files for widgets and functions to promote code reuse