HomeFinder 2.0

Jun Baek and William Taggart

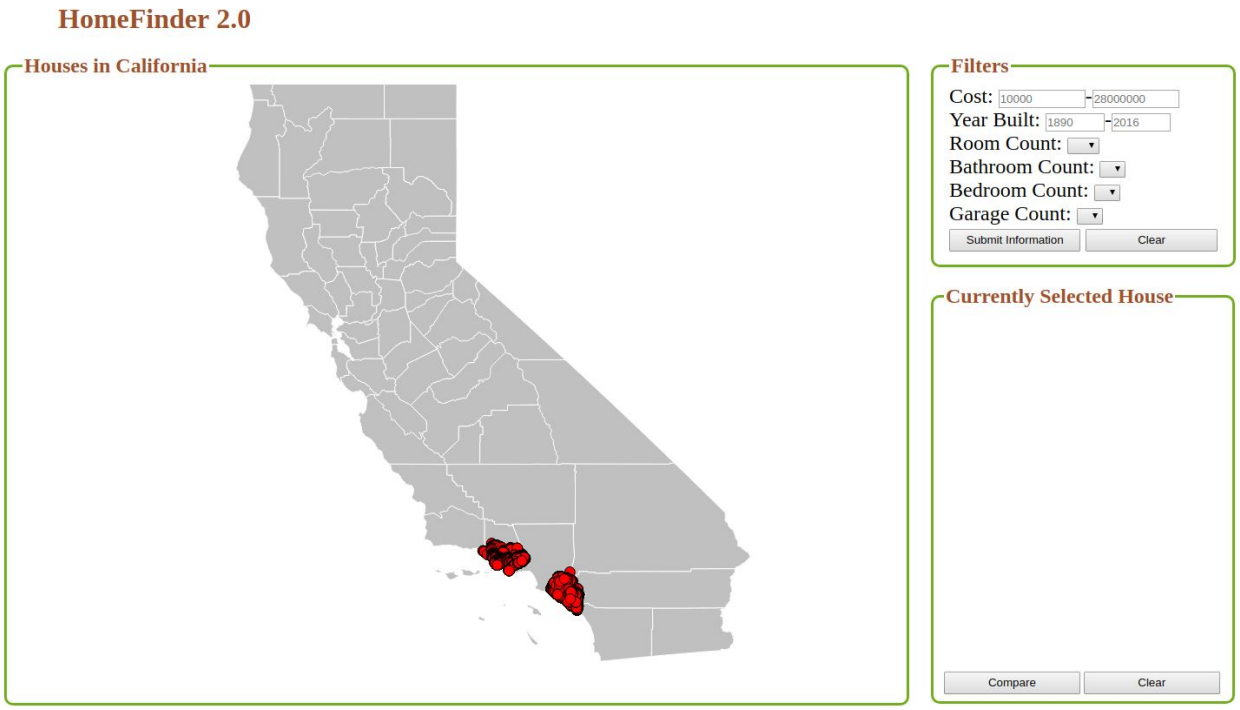

Fig. 1. Full rendering of the HomeFinder 2.0 tool.

Abstract—HomeFinder 2.0 has been designed to improve the original HomeFinder application. By analyzing the initial application and other applications with similar purposes (Home Search and Map visualizations), we were able to pinpoint specific items that needed improvements and identified other functions that will improve the overall home finding experience. Through our application, we hope to provide the user with the most effective and efficient home search experience (focused on a dataset of homes in southern California). We focus on improving the filtering and data analysis capabilities of a home search system by adding more filters that can be found in apps like Zillow, while also increasing functionality by including a comparison feature of homes. As we analyze HomeFinder 2.0, we also recognize that added functionality and constraints can be added to improve the overall usability of certain features such as zoom and panning.

Index Terms—Homes, house search, maps

1 Introduction

HomeFinder 2.0 has been designed to help people in southern California find the right home for them. When we looked at the original HomeFinder application in class, we noticed that the application lacked depth and usability and didnt support important easy to use interactions. One of the first problems that we looked to fix was the ease of filtering. In the original HomeFinder application, the filters used sliders to narrow search results, but they were shown to lack precision and caused frustration with users. When developing our application, we wanted to provide the most precise filters possible (within our dataset), which allows users to directly input their criteria within the ranges our dataset provides for each field.

Another issue with HomeFinder was their use of position filters. The A/B position filters in the original HomeFinder took up a large portion of the filter area and are rather ineffective when thinking about how the average user today interacts with spatial visualizations. With todays technology, users expect to be able to pan and zoom. These two interactions essentially allow the user to filter by position by simply altering the map itself to only show the homes in a certain specified area of the map. We also added tooltips when hovering over a particular home so that it was not necessary to fully select a dot on a map to see any information about it. Since our main function was 'Compare', it would be a lot of extra work for the user if they had to fully select any home they may or may not want to compare.

While many changes and updates were easy to work through, the compare function along with the data collection for this application provided the biggest struggle. While Zillow holds a database of up to date housing information and data that we could have used to make HomeFinder even better, they purposefully keep that data away from student projects. Our dataset was built for machine learning purposes rather than visualization, so some of the data required some tweaking before it was usable in our application.

The largest improvement over the original HomeFinder is our com-

• *Jun Baek and William Taggart are students at University of Maryland.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

pare function. This function allows the user to select multiple homes and compare them. When discussing the needs of home buyers, we realized how important it was to see how two or more homes compared. However, with the original application, there was no easy way to see the information about two different homes side by side. With that in mind, we allow the user to compare multiple homes side by side with important comparison statistics to make the comparison as easy as possible.

2 Related Work

HomeFinder 2.0 draws on the previous work in home searching and map visualizations. Additionally, HomeFinder 2.0 is built using the various tools outlined in other work using d3.js.

2.1 Home Search

Since the internet has spread of every home, the search for a home that used to take people across the country is starting to be done from the comfort of their old living room. HomeFinder [6] was created in 1992 but fell short due to the lack of capabilities and well-formed data relating to the topic. Homefinder provides a reasonable starting point for a home search application that combines a map visualization. Today, companies like Zillow and Homes.com provide online home search services for just about anybody. These options rely on proprietary data that reduces our ability to replicate and truly improve on their functionality and usability, however, they do lack some important characteristics of a home search. Neither application provides a side by side comparison across all features.

While Zillow and Homes.com have created commercial uses of home search, there has been a more recent academic addition to the Homefinder application [8] that details increased functionality and strong user experience. ReACH [8] has created a novel way to identify the goals of a home search. By incorporating a function of reachability, a combination of location and access to important parts of the home searchers life, ReACH allows the user to investigate homes with a level of intricacy that other home search applications have not yet mastered. ReACH also relies on the integration of personal preferences and routines in order to determine the target locations taken into account in reachability calculations to make sure that the tool is optimally customized to any particular user.

2.2 Map Visualizations

Google Maps and many other major map visualizations use the Mercator projections as we have in this project as well. However, these projects aren't the most accurate representations of the earth and the distance between points [5]. In Jenny's work [5], he discusses the issues with the Mercator projection but understands how it has become the standard for web projections due to its use in Google Maps. The Mercator projection is acceptable at a large scale (world view) but falls short in its accuracy at smaller scales. Jenny suggests many other projections that could be used in small area situations such as the Lambert azimuthal projection because it reduces the distortion created by the Mercator projection at these small scales.

Smooth zooming and panning [7] has been discussed as a major issue in our application and has been discussed in great length as a solvable problem in the field of map and other spatial visualizations. Researchers agree that being able to view spatial information at different scales is extremely important to the understanding and consumption of spatial data. However, we relied on d3.js for zooming and panning functions and did not dive deep into the technical aspects of a smooth zoom.

2.3 Interactions and d3.js

Throughout our class, we discussed the different interactions [3] that help the user analyze a set of data through the lens of d3.js [1]. These interactions help the user develop an understanding of a dataset and a system at different levels starting with the base level of view specification (users can narrow the dataset to be more manageable). These interactions grow to incorporate view manipulation where the user can actually influence the view itself. Zooming and panning are the biggest examples of this in map visualizations [3,7,8], which falls in the navigation category. Other work with map visualizations are filtering (a type data/view specification) and, in addition, our work will incorporate aspects of coordination (being able to provide multiple views side by side for comparison) [4].

Throughout our project as well as research and development, we took advantage of one of the largest visualization libraries available [1,2] and the implementation of useful functions, such as zooming and panning mentioned above. D3 has been built to help developers create effective visualizations through one set language and framework [2] and specialize in binding data points to visualizations. We use these functions throughout our project especially when binding data to our dot representations of homes.

3 Design Goals

When designing an upgraded HomeFinder application, we focused first on how to improve the faults of the original application and then to increase the functionality and interactivity of that original application. Heer and Shneidermans taxonomy of interactive dynamics [4] guided our design goals starting with filtering and visualization through recording and annotating with our added features.

TABLE 1: Taxonomy of interactive dynamics for visual analysis

Data & View Specification	**Visualize** data by choosing visual encodings. **Filter** out data to focus on relevant items. **Sort** items to expose patterns. **Derive** values or models from source data.
View Manipulation	**Select** items to highlight, filter, or manipulate them. **Navigate** to examine high-level patterns and low-level detail. **Coordinate** views for linked, multi-dimensional exploration. **Organize** multiple windows and workspaces.
Process & Provenance	**Record** analysis histories for revisitation, review and sharing. **Annotate** patterns to document findings. **Share** views and annotations to enable collaboration. **Guide** users through analysis tasks or stories.

Goal 1. Promoting precise data filtering. We improve on the filters of the original HomeFinder by allowing the user to focus on information that is important to them. The original application [6] forced the user to filter simply on location, price and bedrooms. HomeFinder 2.0 focuses on expanding these filters to provide the user with more options without forcing them to engage with every single filter.

Goal 2. Incremental data dives to increase understanding as interest increases. As users look for homes they start their search by filtering and then viewing their options one by one. In order to find a balance between not enough information and overloaded information, we start by providing the most important information up front, and then allowing the user to interact and gain more information about a particular house and figure out how that house compares to other houses through a series of easy to follow steps.

Goal 3. Improved interactions from HomeFinder [6].

3A. Improving map interactions through zoom and pan. Zooming and panning as a means to filter by location helps us match the users expectations of the application. (zooming panning source)

3B. Select and compare capabilities. Building on Heer and Shneidermans taxonomy of interactive dynamics [4], we need to increase the users ability to select houses and actually access more information. Then increase the users ability to view two different parts of the data side by side (coordination [4]).

4 Methodology

With ease of use in mind, we wanted to make both the filtering and analysis simple and direct.

4.1 Layout

In order to develop an appropriate map visualization for our dataset, we used TopoJSON (an extension of geojson) and geoMercator to create a

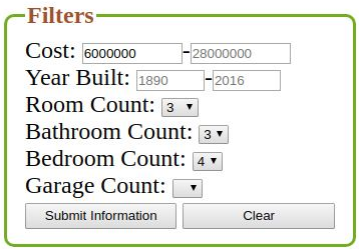

Fig. 2. Direct Filtering System in HomeFinder 2.0

map of California and used GeoJSON to visualize dots to represent the latitude and longitude of the homes in our dataset. Even the best datasets we found had data points that did not fit in the visualization. The dataset had homes that cost 100 dollars and had zero bedrooms, which forced us to filter the data before even letting it enter the visualization.

As discussed in the introduction, the original HomeFinder made an inefficient use of their computer screen, taking up space for location filters and crowding the map with A and B location filters. In our application (Figure 1), the filters take up a very small portion of screen real estate, allowing the user to access more in-depth information about the homes without having to alter the location of items on the screen. It was important for us to maintain that location consistency because the map visualization is meant to change a lot through zooming and panning so we did not change the rest of the application to have to be altered to view the important information.

Many of our design decisions focused around the taxonomy of interactive dynamics [4] described in our design goals. We started with data and view specifications by using the position encodings described above. As we discussed throughout the class, position encodings are some of the most easily distinguishable to the human eye and are additionally relevant when dealing with spatial data such as longitude and latitude.

4.2 Filtering

Our active filters (Price, Room Count, Year Built, Bedroom Count, Bathroom Count, and Garage Count) all allow the user to directly identify which data points to look at. Our inputs allow for a user to find the specific house at a much faster rate than other input methods, but we do realize that they can make it harder to explore the different options. Our design was meant to make it easy to compare multiple similar homes of a specific type, rather than sift through a large set of homes. Because of this design focus, we moved away from the sliding filters of the original HomeFinder application (despite their usability for quantitative features). We decided upon direct inputs because they allow for more targeted analysis of the dataset rather than the general and exploratory function of a slider. With our target demographic being home buyers, we are expecting them to already have a certain type of house in mind (budget, size, etc.) and our system would make it quick and easy for them to directly input these preferences and get right into the home search.

4.3 View maniupulation

Following into the next section of Heer and Shneiderman's taxonomy, we look at our implementations of view manipulation with selection and navigation.

4.3.1 Selection

Following our design goal of incremental data access, the first steps are tooltips and selection. In order to help users understand the simple dots on a map, we implemented tooltips that include the most important information about a potential new home. By taking into account the features in our dataset and the most important deciding factors according to new home buyers, we decided on the most important features (Price, Bedroom Count, and Bathroom Count) which fill up our tooltips. These

Fig. 3. Information available once a home is selected

tooltips are meant to help the user decide if they want to learn more information about a particular house and then the user can click on the dot/home and view more information (Figure 3). We found it extremely important for the location to still be recognized once a particular home had been selected which led us to our next encoding, the color encoding of a selected home. The color encoding allows the user to view the specific home information (Garage Count, Room Count, Pool Count, etc.) apart from the map while still keeping track of the house's location via the map.

4.3.2 Navigation

The original HomeFinder application relied on position filters, but since then people have become more and more used to zooming and panning on interactive map interfaces such as GoogleMaps. Our application implements basic panning and zooming functions while allowing the user to filter their visualization by location (Figure 4).

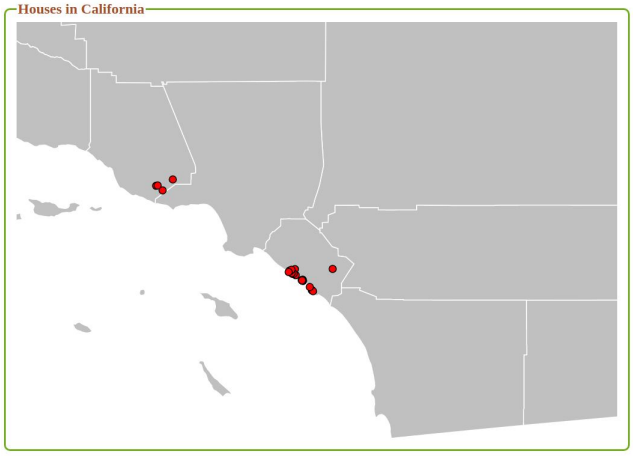

Fig. 4. Map interactions are made easy by using the mouse to pan and zoom.

4.4 Compare Function

Our biggest improvement over the original HomeFinder application was our implementation of a compare function. The function allows the user to engage in the other levels of view manipulation (Coordination and Organization). The compare function allows the user to access the information for multiple different data points and look at their features side by side. When we discussed what home buyers would like in an

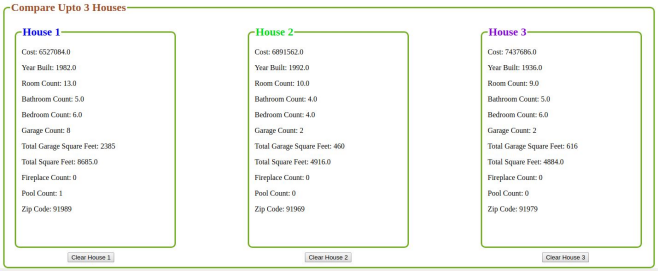

Fig. 5. Example Compare Functionality

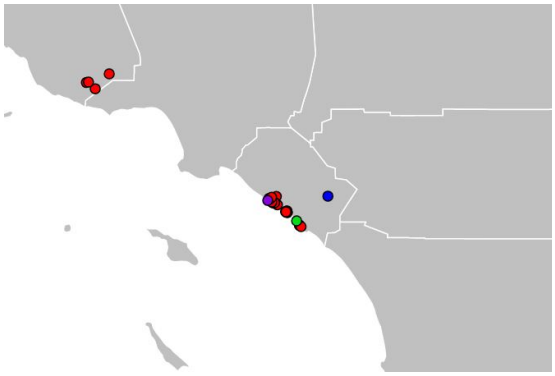

Fig. 6. Multiple colored data points used to identify the homes being compared

application, we understood that with the amount of different factors that go into the decision it was important to make sure that the users aren't forced to remember all of the information for a particular home. Our compare function allows a 3-way comparison for two main reasons: **1.** The design of the 3 info boxes allows the user to compare 3 at once without having to scroll, **2.** The time constraints of the project made it more reasonable to make the one extra comparison than juggling the design constraints of adding too many additional comparisons. (see Figure 5)

4.4.1 Initial Compares

If the user has already selected a home in the selection box, then that home will automatically fill the 'House 1' section of the compare function when the button is clicked. We decided that if the user had decided to click the compare button (directly underneath the information about the selected home), then they were most likely trying to compare that house to other homes. In the case where there was no previously selected home, the program will wait for a new home to be selected before filling the first section.

4.4.2 Colored Compares

When discussing selection, we realized that the information would be lost if we did not maintain a marker of the selected home's position, so we added a color encoding to identify the selected home. With the compare function, we decided to implement a similar color encoding to identify up to 3 homes being compared. In order to encode the location of the homes being compared, we decided to match the color of the dot to the color of the text in the compare boxes. This allows the user to maintain an understanding of the location of each home as they compare the rest of the features of each home. (see Figure 6)

4.4.3 Selecting Additional Homes

When comparing homes, there is a chance that a user will want to compare a home with another home that may not be visible on the map due to the filters in place. We needed to make sure that refiltering the dataset did not disrupt the currently selected homes. For example, if the user selects 'House 1' at 500000 dollars and then resets the filters to only include homes that cost more than 600000 dollars, the visualization will keep 'House 1' on the map in order to support the comparison by location as well as in the other features.

4.4.4 Clearing Compared Houses

Our compare function also allows individual homes to be cleared so that a clear function does not clear all of the houses at once. The goal is to allow the user to easily swap out homes for other homes. In doing this, we allow for the user to easily correct human errors (clicking the wrong dot – extremely important with overlapping dots) and allow the user to continue to compare their favorite homes to a wide variety of homes without having to find it on the map again. It was important to us because we nixed a save function in the name of time, but we still thought it was important to make sure that the user didn't lose a particular home in the vast amount of data.

5 Evaluation and Results

First, we wanted to see how people interacted with our application without any prompts to see if the design spoke for itself. With all of our subjects, the lag created by zooming and panning caused a reaction in the user that led to them being irritated with the application. One user decided to try clicking everything while the map was trying to load and ended up slowing the application down even more. However, these negative reactions always seemed to happen at the beginning of the interaction sequence and, once the user had filtered the data to become specific to their goals, they were able to utilize the application to its full potential.

5.1 Find the most expensive house

After they had the chance to explore, we asked them to try and find the most expensive house (timed and given two minutes).

User 1.
Successful given the timeframe of two minutes. This user focused their filters solely on the price of the house, and when they reached a small enough dataset with only a small set of the most expensive houses, they used the compare button to quickly access the details for each of the possible most expensive houses and found the answer.

User 2.
Unsuccessful given the timeframe of two minutes. This user attempted to zoom and pan the data without filtering the data first which caused a delay in the system. Once the application was ready, the user filtered the data by price attempting to look at the houses with a price above 1 million. This dataset was still so large that it caused lag on zoom and delayed the process once more. The user then filtered above 15 million which greatly reduced the remaining data. Towards the end of the two minutes, the user relied on the select function and not the compare function in order to look at the differences between the different houses. It seemed that the user would have been able to find the most expensive house had they been given more time (or if we had been able to fix the lag in the system).

User 3.
Unsuccessful given the timeframe of two minutes. This user had one fatal flaw. The user attempted to fill every filter before submitting the form. Instead of simply filtering by price, this user added a criteria for each of the categories at each iteration of their filtering. While it allowed the user to reduce the shown dataset to a small number of points, it did not allow the user to actually see the most expensive houses due to restriction on other categories, such as Year Built. This user used the compare function to directly see a side by side price comparison for the different houses, but did not successfully find the most expensive house because of the other filters that restricted their search.

5.2 General Results

- Every user understood that they could select a house by clicking on it and that they could pan across the map using the mouse.
- However, the scroll to zoom function that was implemented did not always have the expected effect. Once the user clicks on the Compare button, the added objects below the map create a need to scroll down the web page, but this scrolling would then interfere with the maps zoom function, which would slow down the application and also throw off the user.
- While we emphasized the ability of our filters to focus on specificity, this level of specificity was not helpful when users first engage with the data and need a bit of time to explore the data generally before developing a goal/task.

Possible fixes to the issues discussed in this section will be discussed in the following section.

6 Discussion and Future Work

After evaluating HomeFinder 2.0 through the lens of a small case study, we can tell that while our application created an improved version of the original application, it did not always create the expected interactions that we had hoped.

6.1 Limitations

- When we first started our application, the first issue we encountered was the lack of data available to implement a HomeFinder application. With services such as Zillow and other commercial home search applications, they tend to keep their data close to home to avoid someone replicating their business. We were able to find this focused dataset through a data science competition, however, the data lacked some of the properties that we had initially hoped to incorporate in our project.
- Once the data was secured, we looked for an effective way of representing the data in a map form and we found a useful graphical representation of California. Since we were focused on different interaction features, we did not take the time to ensure that this representation of California and the algorithms used to manipulate it would be able to create a seamless interaction. While we focused to create an effective compare function, the map itself left something to be desired. With more time, we would have been able to investigate the best options to increase the speed of our zooming and panning. One thought would be to reduce the number of data points shown (so many are being overlapped at any given time) when at the lowest zoom level and as the user zooms in, the visualization would slowly add the other data points.
- With the time constraints of the semester in mind, we knew that implementing a massive compare function would be beneficial, but would not work for our timeframe. We settled on a 3 home compare function that uses color codes to identify the home on the map and in the information box. Our function allows the user to see the homes side by side and compare them, however it relies on users to look through each property and compare on their own. Given more time, we could have used styling techniques to highlight the lowest/highest price of the houses being compared (and similar techniques with other categories).

6.2 Future Work

- At the beginning of our development process, we hoped that zooming and panning would be a huge improvement over the original HomeFinder. However, due to the lag and our systems reaction to the data, it was hard to make the zooming and panning as smooth as we had hoped. While there may be ways to alter the data and develop a smoother zoom and pan, we also think it would be helpful to allow the user to switch zoom and pan on and off. We suggest implementing buttons that toggle these interactions on and off when the user wants to use them or not. This way the user would no longer attempt to scroll down the webpage and accidentally zoom out of the map, and any small mouse actions would not have the immediate effect of lagging the entire site.
- While our solution above could improve usability and performance, we would want to improve the data and functions we used to implement the zooming and panning, as well as the map itself. When analyzing the points of improvement that were identified through our evaluation, the lag created in our map visualization caused the most frustration in our users. This would have taken much longer to analyze and optimize the speed of our application, but with some additional time, this problem could have been addressed effectively.
- One aspect that we were unable to implement in our application at all was the aspect of home images. We realized that it was extremely important for a homebuyer to see the home before investing time or energy into pursuing the home. We had discussed this aspect of the project at the beginning but pushed it to the side because it would have required us to go a bit off track from our design goals. Once a home is selected, it would be beneficial to include a link to google maps images of a home (or embed images from google maps) so that the user could have a visual of the home to consider.

7 Conclusion

As we see that HomeFinder 2.0 is not a perfect application, we have made sure to recognize the areas for improvement that have been identified above. Throughout the process, we found it helpful to focus on increasing the breadth of the applications across the taxonomy of interactive dynamics [4] and focusing on specific interaction goals. We were able to accomplish certain goals (comparison and improved filters), while other goals fell short (problems with the speed and smoothness of zoom and panning). Future goals include, improving the style and design by adding images of the homes similar to Zillow and adding other technical features similar to ReACH [8]. With additional time to work on the project we would be able to provide the user with the most effective and efficient home search experience (still focused on a dataset of homes in southern California).

Acknowledgments

Team Member Contributions: Jun Baek contributed more to the coding of the application. William Taggart contributed slightly more to the writing of the report. Distribution of work and communication was well accomplished through a communication app called GroupMe. The authors wish to thank Ben Shniederman and Jeffery Heer for putting in so much work in this area.

[1–8]

References

[1] M. Bostock. *D3.js - Data-Driven Documents*, 2019.

[2] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. 2011.

[3] J. Heer. Cse512-interaction. 2019.

[4] J. Heer and B. Shneiderman. Interactive dynamics for visual analysis. *ACM Queue*, 10(2), 2012.

[5] B. Jenny. Adaptive composite map projections. *IEEE Transactions on Visualization and Computer Graphics*, 18(12), 2012.

[6] B. Shneiderman and C. Williamson. The dynamic homefinder: Evaluating dynamic queries in a real-estate information exploration system. 1992.

[7] J. J. van Wijk and W. A. Nuij. Smooth and efficient zooming and panning. 2003.

[8] D. Weng, H. Zhu, J. Bao, Y. Zheng, and Y. Wu. Homefinder revisited: Finding ideal homes with reachability-centric multi-criteria decision making. 2018.