Places: Spatiotemporal Comparison of Location Histories

Matthew Rafuse

Fig. 1. Example usage of Places, comparing three distinct time spans. The map shows seven locations visited in the selected time spans, with their size reflecting the duration spent in each location. The Gantt chart shows a significant overlap between two time spans in Halifax, with a few points in which all three time spans were in Halifax.

Abstract—In the modern day, thanks to mobile devices, users are presented with a wealth of information about their own histories and the histories of their close friends and loved ones. In particular, with the advent of consistent location data collection, rich location history datasets have emerged. These datasets allow users to review and reminisce on where they have been, and the combination of the timelines of multiple people can permit them to compare their histories. In this paper, I present the application *Places*, which visualizes and allows comparison of spatiotemporal data. The application uses two views to visualize spatiotemporal data - one spatially, and one temporally. This application aims to simplify the process of comparing location histories.

1 Introduction

As our modern devices and related services collect more and more data about our lives, users are presented with a wealth of information about their own histories and the histories of their close friends and loved ones. In particular, with the advent of consistent location data collection through services provided by the likes of Google, rich datasets have emerged. These datasets allow users to review and reminisce on where they have been, and the combination of the timelines of multiple people can permit them to compare their histories.

Visualization of spatiotemporal data is a well-researched topic in Information Visualization. However, many modern applications are concerned mainly with scalability, allowing users to view, understand and explore larger and larger datasets. Due to this, these applications often aim to capture and display general trends in the data, instead of focusing on the particularities of a single point in the data, or the relationship between a small number of these points. An interesting avenue of research is to attempt to incorporate such an approach to understanding and comparing user-level location data. In this paper, I present an application that allows users to explore their own location timeline both in isolation and in comparison to the location timelines of loved ones or close friends.

This application, *Places*, visualizes spatiotemporal data collected by Google, downloaded via Google Takeout. The application consists of two main views - the Map, and the Timeline. The Map presents users with a high-level overview of the location and duration of stays in the given time span. Multiple closely related points are grouped together into stays, each of which is shown as a magnified view of the overarching location of the stay. The Timeline presents users with the chronology of their stays. The Timeline is a modified Gantt chart that shows the order and duration of each stay, giving the user a sense of the chronology of the stays.

Additionally, an important feature of *Places* is the ability to compare and contrast individual time spans. For example, in Figure 1, it is clear that both the teal and purple time spans spent the bulk of their time in Halifax, while the green time span is far less centralized. The user can then conclude that it is far more likely the timelines of teal and purple overlap than that of, for example, teal and green.

The rest of the paper is structured as follows. Section 2 outlines the motivation for this application. Section 3 summarizes related works and discusses how they compare to *Places*. Section 4 discusses the design of the application and the motivation surrounding choices made in the design. Section 5 outlines the implementation of the application. Section 6 explores a use case study of the application. Section 7 outlines some limitations of the current application and potential future work.

• *Matthew Rafuse is with the University of Waterloo.*

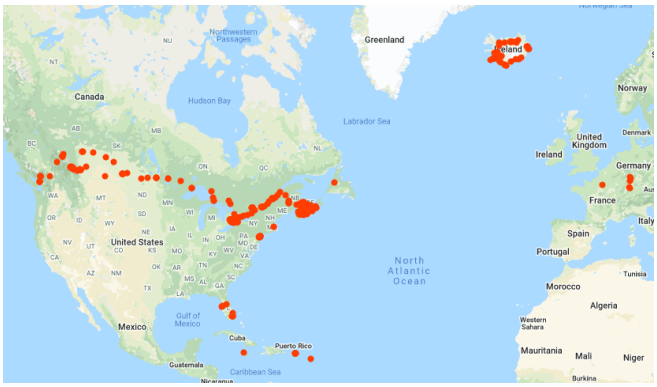


Fig. 2. The default view of Google maps timeline. The points are all placed on the map, regardless of when thy were visited, making it impossible to distinguish at a glance when a user went where.

Fig. 3. Visits is an application that aims to allow users to explore their personal timeline, with the size of the bubble corresponding to the duration of the stay. It supports a number of interactive features, like zooming into a particular section of the timeline.

2 Motivation

In the age of "Big Data" there's a wealth of works on large scale, efficient visualization of data collected from thousands of subjects. However, in this rush, these solutions often end up not being particularly well suited for small-scale visualization and comparison, making it difficult for normal people to review their own location data.

This is the motivation for Places. The aim with Places is to create an application that accurately visualizes location data from a small number of sources as accurately as possible. This allows users to reminisce about their own histories and vacations, and compare their histories with those close to them, or even with other points in their own life. Existing works either are built to compare thousands of sources, or do not support comparing sources at all, and are built only for a single source.

3 Related Work

There has been a wealth of research done on representing spatiotemporal data. Bach et al. assembled a survey of such techniques, but we will focus mainly on those relating to location data [4]. There are five applications we will review in this section.

3.1 Spatially Focused Representations

Google Maps Timeline [1] shows all visits across the lifetime of the user, each as a point, with no distinction temporally. This allows a high-level idea of where a user has visited, but completely removes any sense of how the points are related temporally, and the duration of the stays. Comparison of location histories is not supported. If such a feature was implemented, the approach taken would not lend itself to comparison outside of a high level overview - i.e., both users visited Paris at some point in their timeline.

Thudt et al. present Visits [8], which aims to fill a similar niche as this proposal. Visits uses map-timelines to show the trips in a given time frame, with the size of individual bubbles representing the amount of time spent there. Many of the design choices of Visits can be seen in Places, particularly the concept that width corresponds to duration. However, their approach cannot easily be used to compare multiple people, since comparison of map-timelines spatially is limited by the fact they trade spatial accuracy for a clearer depiction of the chronology. By combining the geographic and temporal data into a single map-timeline, comparison of two timelines for overlapping stays is difficult. In Places, the addition of the Timeline that takes the form of a modified Gantt chart allows a user to easily compare the location histories of multiple people, and focus in on similarities and relationships between users.

A follow-up to this paper looks at using similar techniques for creating visual mementos to share with friends [9]. The project shares a

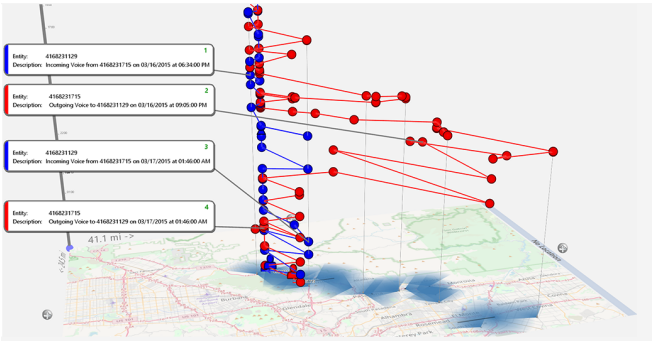


Fig. 4. GeoTime takes a novel approach, representing the passage of time using the change in the z-axis. This allows a general idea of how subjects moved around on the map.

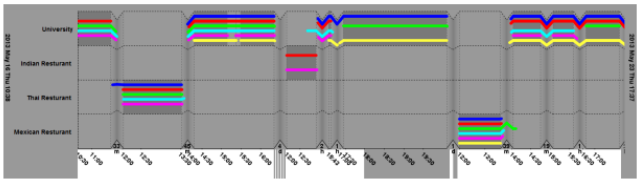


Fig. 5. MovementSlicer represents locations as rows on a Gantt chart, with users jumping discretely between locations. This allows users to, at a glance, determine when subjects were and were not in the same location.

[1] Google Maps Timeline - https://www.google.com/maps/timeline

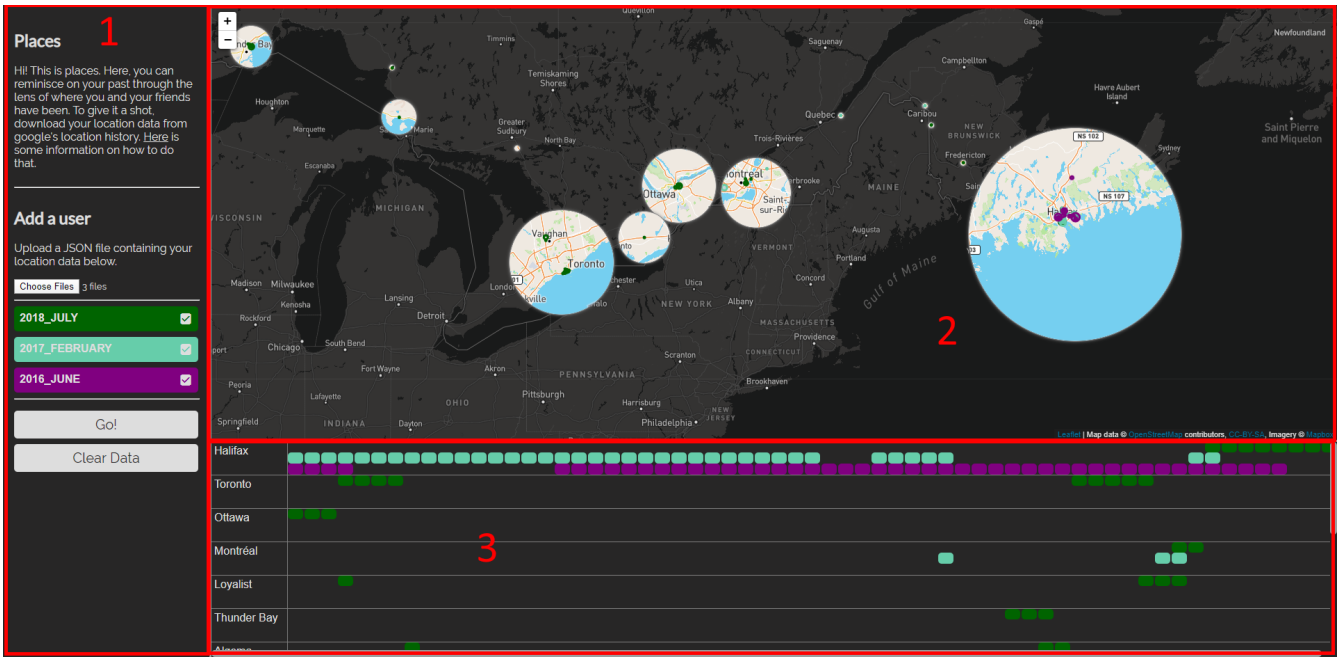


Fig. 6. An annotated view of the Places interface. **1** corresponds to the sidebar, **2** corresponds to the Map view, and **3** corresponds to the Timeline view.

fair number of similarities with Visits, particularly in its use of map-timelines, which as discussed are not optimal for comparison of location data. Additionally, their application requires users to know exactly what trip to save, and does not easily permit exploration of the user's location timeline.

Kapler and Wright present GeoTime [7], which shows data from multiple users on a map temporally, with the z-axis encoding the temporal dimension. GeoTime is used for presenting data for use by law enforcement, and so allows law enforcement to track the movements of suspects, victims and bystanders, comparing their relative distance and locations. This approach works well over short time periods (< 1 day), but does not easily scale to longer time periods, as the usage of the z-axis becomes cumbersome. Additionally, this method does not easily encode the duration of a user's stay in a given location.

3.2 Temporally Focused Representations

In addition to the spatial representations, there is an important existing work that represents location data in a more temporal way. MovementSlicer investigates improving Gantt charts for spatiotemporal location data, and comparing the movements of several people over some time [6]. However, their Gantt charts do not easily represent the actual physical locations of the users, an issue solved by the dual views of Places.

4 Design

Places can, at a high level, be broken down into three distinct sections, as outlined in Figure 6. These three sections are the Sidebar, Map view and Timeline view. This section will summarize the design and functionality of each section.

4.1 Sidebar

The sidebar is the entry point of the application, shown in Figure 6 as box 1. It provides a basic explanation of the application, and prompts the user to provide a set of location history files. This file must be a JSON file formatted to match the style of google's exported location history. Each submitted file, up to a maximum of ten, is provided a unique color and the name is set to the file name. Each file can be individually enabled or disabled via a button on the right hand of the corresponding entry in the list. Once the user has included all the desired files, they press the "Go!" button which triggers the computation of data to be shown in the Map and Timeline views.

4.2 Map View

The Map view is one of the two main views making up the application, shown as box 2 in Figure 6. There are three main components to the Map view. We will discuss each in order of granularity.

4.2.1 Global Map

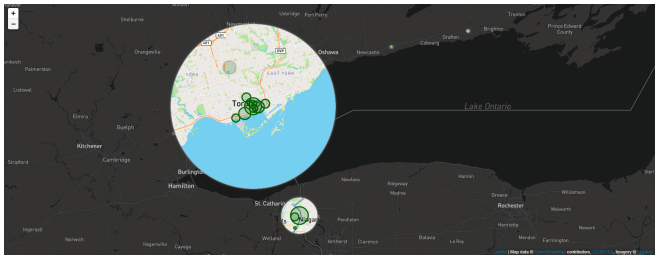
The global map is the backdrop of the view, providing context for the location of each stay in the world. In order to provide sufficient contrast with the stays, the map is drawn in dark colours. The global map supports click+drag to move the view box, and zoom to focus on particular areas or gain a macro view of the locations visited. In order to increase the efficiency of the applications, stays not in view are not rendered. When the view box changes, this triggers and event that causes the application to recompute the stays and associated information.

The zoom level of the global map is an important variable in these computations. The higher the zoom level, the more granular the data is. This can be seen when comparing Figure 6 and Figure 7(a). The stay that previously contained the entire Golden Horseshoe area has split into multiple stays. This has knock-on effects on the Timeline View, as is discussed further in Section 4.3.

4.2.2 Stays

Distributed across the Global Map are *Stays*. These are groups of visits that are all in a similar location. These points are not necessarily from the same timeline (if multiple are loaded), or the same time span - i.e., if a user visited Halifax at the beginning and end of their time span, both sets of dots would be shown in the Stay.

Each stay appears as a brightly coloured, magnified view of the general area of the similar visits. The exact location of the center point of the stay is set as the mean location of all points in the stay. To visualize the amount of time contained in the stay, the radius of the circle changes proportionally to the duration. Clicking on a stay will magnify the stay, doubling its size, allowing users to look more closely at a stay without needing to zoom in or otherwise manipulate their current view box.



(a) An example of a zoomed-in view of Southern Ontario. Note that compared to Figure 6, the stay centered on Toronto has split into two separate stays - One centered on Toronto proper and one centered in Niagara Falls. It is also clearer how each point in the stay is drawn.

(b) An example of a highly local view of downtown Toronto. The magnified map representing the stay has disappeared, and instead each point in the stay is drawn directly on the global map. Similarly to stays, we see that each point has a width corresponding to the time spent at the point.

Fig. 7. Two different zoom levels, showcasing the dependence of the method of visualization on the current zoom level. At higher zoom levels, granularity increases, allowing users to narrow their view to locations they are particularly interested in.

At higher zoom levels, the stay becomes redundant and is no longer shown. Instead, the set of points that make up the stay are drawn directly on the global map, as shown in Figure 7(b).

4.2.3 Points

The ”point” is the atomic data structure in Places. It represents a single location at a single point in time. The point is centered directly on the location is represents, and its width corresponds to the time spent at that point. The color of the point corresponds to the time span it originated in, and is semi-transparent to allow comparison of points centered at the same location. Similarly to a stay, clicking on a point will double the size. This is implemented solely so points behave identically to stays. Depending on the zoom level, points will be drawn either in the map representing the stay they are apart of, or the global map directly.

4.3 Timeline View

The Timeline view is responsible for giving the user a sense of the chronology of the stays in the Map view. There is a one-to-one correspondence between the rows of the timeline and the stays visible on the map. The rows are sorted in descending order by the total time spent in each corresponding stay, and their names are derived by querying the MapBox API [2] for location data at the location of the stay. Depending on zoom level, the location data will either be more or less granular - for example, at a low zoom level, the general region or city name will be used, while at a high zoom level, road names and addresses will be used instead.

The columns of the Timeline view represent 12 hour intervals. If, at any point in those 12 hours, time was spent at the specific stay or point represented by that row by a particular source, an entry for that source will be added to the cell. Clicking on a specific entry will scale up the corresponding stay or point, and double clicking the entry will cause the camera to focus on that location.

4.4 Interaction

As mentioned in Sections 4.2 and 4.3, there is user interation in the Map and Timeline views, as well as interaction between them. In the case of the Map View, users can reorient the viewbox by clicking and dragging as well as zooming in and out, modifying the lens through which the data is presented - increasing and decreasing granularity, and filter what data is shown. The Timeline view allows users to navigate through the selected sime spans, and clicking on active entries in the Gantt chart will highlight the data in the Map view. Additionally, when double clicking an entry in the Gannt chart, the Map view's viewbox will change to zoom in on the particular data for that entry, and filter out other, extraneous information.

5 Implementation

The application is written in TypeScript [3]. To render the map, leaflet [1] was used. D3.js [5] is used to implement the timeline view due to it's intuitive way of working with data. In order to ensure the privacy of its users, no data is sent to the host server - all computations are done locally on the user's machine. The application runs in three phases- initialization, rendering, and cleanup. The application initializes on first load, setting listeners on UI elements and preparing the file parsing functionality. It also parses the user data. An important note is that currently, all data is transformed to be relative to the earliest time in the data given. That is - the columns are not absolute dates, but their relative distance from the start time. This is to allow users to compare time spans from different points in their own timeline - this can be seen in the images provided, since the three example timelines are all from different points in my life and not simultaneous.

Algorithm 1 The algorithm used to build stays out of sets of points.

```
/* Dependencies */
P: The set of time spans to render.
S ← {}: The list of stays to render.
d: The cutoff distance for inclusion in a stay. Depends on the zoom level.

s_cur ← {}: The current stay.
for each p ∈ P do
    for each p_i ∈ p do
        if s_cur = ∅ or distance from mean(s_cur) to p_1 ≤ d then
            s_cur = s_cur ∪ {p_i}
        else
            S = S ∪ {s_cur}
            s_cur ← {}

S_final ← {}: A filtered version of S with no overlapping stays.
for each s_i ∈ S do
    if ∃ s_j ∈ S_final s.t. distance from mean(s_j) to mean(s_i) ≤ d then
        s_j ← s_j ∪ s_i
    else
        S_final ← S_final ∪ {s_i}
return S_final
```

Once the user hits ”Go!”, the application enters the rendering phase. In this phase, the application iterates through all supplied data, and groups individual points into stays. It does this using a process outlined in Algorithm 1. Once all the points have been grouped into stays, the stays are rendered onto the screen. The final phase is cleanup. This phase runs directly after interaction from the user - be it zooming in or out, or moving the viewbox. The cleanup phase iterates through all stays, removing the stays from the map and preparing to run the rendering phase once interaction has ceased.

The application employs a number of data structures in order to visualize the data, the most important of which is the Stay structure. The Stay structure is a logical grouping of similarly positioned points, and is used to keep track of and render stays and points on the map. Each stay is composed of some number of points, and contains much

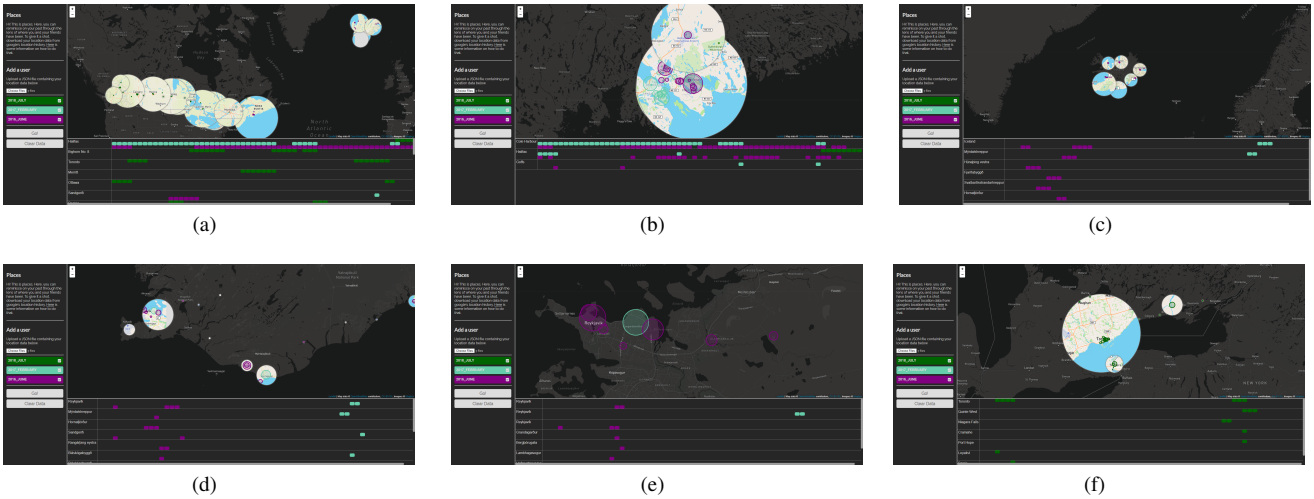


Fig. 8. A step by step exploration of user data in Places. The user first looks at the area of largest overlap, Halifax, before moving on to the vacation of interest in Iceland. Finally, the user decides to investigate where the third time span was a the same point in time, looking at Toronto.

of the logic for rendering stays and points. Each point, representing a location on the map and duration of time, also contains an id that is used as a reference for the timespan it corresponds to, and is what is used to set the colour of the points on the map.

6 Scenario

In this section we will break down an example use case of the application. We will compare three time spans and see what insights can be gained. The first step is to load in the three time spans. The three time spans we are using are three months of my life - July 2018, February 2017, and June 2016. For the purpose of the study, I will refer to each time span as an individual person - July, February and June.

The first step is shown in Figure 8(a). Directly after pressing "Go!", the user is shown a high level overview of the locations of the three people. Predominantly interested in the area of largest overlap, the user quickly surmises June and February both overlap for much of their time in Halifax, and double clicks on the entry to zoom in, resulting in Figure 8(b). The user notes that both June and February spent time at the Halifax International Airport, and realizes they may have gone on a trip together.

Zooming back out, the users scrolls through the timeline and notes that both June and February have stays in Iceland. The user realizes this must be the trip they went on, and zooms in on Iceland, resulting in Figure 8(c). At this point, the timeline makes it clear that the two of them did not take the trip together, as there is no overlap in the Timeline view. Out of curiosity, the user checks where they both visited, and notes there are two locations both visited, as shown in Figure 8(d). The user zooms in on the larger of the two, Reykjavik, and surmises both spent time in Reykjavik, as can be seen in Figure 8(e).

Finally, the user decides to investigate where July was at that point in time. They see time was spent in Toronto at that point, and zooms in to look, as shown in Figure 8(f). The final bit of interesting information notes is that July was in Southern Ontario twice, and the first time did not visit Niagara Falls.

6.1 Dicussion

This example scenario highlights a few of the possible avenues for exploration in the application. The first flow is temporal in nature - the user is interested in time spent in overlapping locations. Exploring this interest leads them to another avenue - reviewing the location data points to a trip to the airport, implying travel. This piques the user's interest and they use the map to look at potential vacation spots, aided by the Timeline.

Places allows users to compare data not only spatially but also temporally, and easily allows switching between the two visualizations.

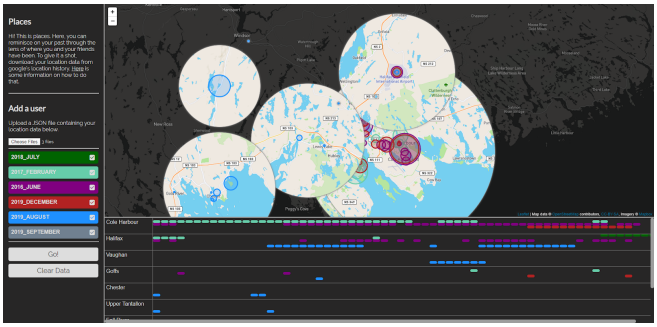


Fig. 9. Viewing large amounts of data centered in a single place can overwhelm the visualization easily, hampering efforts to gain insights about the data.

This allows users to explore their data in ways not easily supported by existing works.

7 Limitations and Future Work

While the current iteration of Places is quite promising, there are a number of usability issues and polish that will improve the user experience that should be investigated. Perhaps the central issue is the fact that the application does not scale particularly well, as can be seen in Figure 9. A number of solutions have been considered, but two main changes would likely make this easier to understand. Firstly, circles should not overlap. We could do this by circle packing - placing all overlapping circles into a larger circle, and centering that circle in the average location of all the contained circles. This sacrifices a minimal amount of location fidelity for a much more readable view. Secondly, the thresholds at which stays are combined can be tweaked so that stays do not overlap - this is less desirable as it would not fix the issue of longer (therefore larger) stays and shorter (therefore smaller) stays overlapping easily.

Another issue is performance. The application is not well optimized and does not reuse rendered maps. Better caching and limiting the number of new map initializations would do wonders for performance, allowing far more data to be rendered on screen efficiently.

The final major limitation of the work is the ability to focus on particular points in time. Currently, there is no method through which a user can narrow their exploration to a particular time frame, i.e. a vacation. The data is shown for the entire time span and the user must search out their locations on the map to narrow their view. Expanding

the functionality of the Timeline view to select sections of the timeline similar to functionality seen in video editors would lead to a large improvement in usability.

In terms of future work, there are a number of features to look at:

1. Automatic downloading of user location data through their google account. This way, users would be able to use the application even more easily, no longer needing to use google takeout to download their data.

2. Allow parsing of formats other than google's location data. This would also expand the ways users can use the application.

3. Allow users to specify the time span represented by a column in the timeline view. This is currently set to 12 hours, which is perfect for approximately month-long time spans, but useless for year-long or day-long time spans. Setting this value would allow users to decide exactly how to use the application.

4. Explore alternate use cases for the application. One idea in particular is to compare the location data of a convicted criminal and suspected accomplice, to extract points in time they were in contact.

5. Allow exploration of data in absolute times as well as relative times. That is, instead of comparing based on different starting points, compare the data based on their exact date, to allow sections that do not overlap.

Places provides a rich set of possibilities for expansion, while already being fairly usable today.

8 Conclusion

In this paper, I presented Places, an application I created to compare and contrast location histories in a spatiotemporal manner. The application uses two views, the Map view and the Timeline view to allow users to explore their histories and compare time spans both in terms of physical location and in terms of time. This paper discussed the design, implementation and a number of use cases for the application, as well as discussion on these results. Places aims to permit users to explore and reminisce on their past, and compare their histories with those of close friends or loved ones, or even to other points in their own timeline.

The source code for this project is available at `https://github.com/Halo4356/Places`.

References

[1] Leaflet — an open-source JavaScript library for interactive maps. Library Catalog: leafletjs.com.

[2] MapBox API. Library Catalog: docs.mapbox.com.

[3] TypeScript - JavaScript that scales. Library Catalog: typescriptlang.org.

[4] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A review of temporal data visualizations based on space-time cube operations. p. 19.

[5] M. Bostock. D3.js - data-driven documents. Library Catalog: d3js.org.

[6] S. Gupta, M. Dumas, M. J. McGuffin, and T. Kapler. MovementSlicer: Better gantt charts for visualizing behaviors and meetings in movement data. In *2016 IEEE Pacific Visualization Symposium (PacificVis)*, pp. 168–175. IEEE. doi: 10.1109/PACIFICVIS.2016.7465265

[7] T. Kapler and W. Wright. GeoTime information visualization. 4(2):136–146. doi: 10.1057/palgrave.ivs.9500097

[8] A. Thudt, D. Baur, and S. Carpendale. Visits: A spatiotemporal visualization of location histories. p. 5.

[9] A. Thudt, D. Baur, S. Huron, and S. Carpendale. Visual mementos: Reflecting memories with personal data. 22(1):369–378. doi: 10.1109/TVCG.2015.2467831