Smart City Visualisation
Smart Taxi Vis: an Interactive Data Visualisation for New York Taxi Operational Data

Qiru Wang
689404@swansea.ac.uk

Department of Computer Science
Swansea University, United Kingdom

Abstract

70 per cent of the world's population will be moving to cities by 2050, this challenges the existing ways of governing city's resources. Smart City, namely the integration of information & communication technology into governances, is seen as one of the most effective way to achieve that. The data generated from information & communication technology carries valuable insights that can be utilised to optimise city's current operation, e.g. improve infrastructure efficiency, enrich healthcare quality or reduce traffic congestion. Data visualisation conveys abstract data into meaningful graphic representations, which is an effective way that helps in analysing the data collected.

Smart Taxi Vis aims to develop a data visualisation toolset that helps extract meaningful information, insights and patterns from the taxi operational data recorded by the New York City Taxi & Limousine Commission. Smart Taxi Vis was developed using MySQL and Java for backend data storage and pre-processing and a web application developed in HTML 5 and JavaScript for frontend visualisation presentation. It provides interactive features to dynamically visualise taxi pickups, taxi dropoffs, average trip fare and average trip distance, delivers the effectual visualisations to unveil useful information such as human mobility patterns hidden in the dataset.

Project Dissertation submitted to Swansea University
in Partial Fullment for the Degree of Master of Science

Swansea University
Prifysgol Abertawe

Declaration

Statement 1

Statement 2

Statement 3

Contents

1 Introduction

Pressure are being applied on the existing way of governancing cities, 70 per cent of the world's population will be expected to move into cities by 2050 [Uni14]. The rapid expansion of cities mandates the implantation of Smart City, the definition of Smart City may vary from country to country, but the common interest behind is to improve life quality of its citizens by the optimisation of current resources [LR12], to mitigate the problems produced by rapid growth of population (including ageing population) and rapid urbanisation [Cho+11]. In every city, operational data of the city is being collected via Information and Communication Technology (ICT) [DA11], while investing insufficient effort into gathering useful insights and knowledge that the data hold [AA08]. One of the reasons behind is that raw data is laborious to analyse with human eyes. With effective data visualisation techniques, which extract the meaningful information by transforming raw data into graphical representations, conveys more information via human visual cognition [WGK10].

Since sight is one of our key senses for information understanding, the initiative of this project is to develop an interactive visualisation toolset, Smart Taxi Vis, revealing the meaningful insights underlie the taxi operational data of New York City, collected by New York City Taxi & Limousine Commission. Smart Taxi Vis is beneficial to users across different levels, the performance of Smart Taxi Vis is set to be a key benchmark for its success, the target was to deliver a smooth and lightweight toolset with cross-platform compatibility, thus the form of web application was chosen. During the planning phase, similar works were reviewed to identify their advantages and disadvantages, and those were thoroughly considered for development of Smart Taxi Vis.

One of the challenges encountered in this project was the data itself. Having huge datasets from different data source providers has resulted in different data formats and more discrepancies, that often require tedious data pre-processing. In order to overcome that, a MySQL database is used in combination with a Java program written, specifically for pre-processing data into JSON format, which is both human readable and programming friendly. Another challenge was the tight schedule, some trade-offs in user interface design and performance optimisation were made in order to ensure the prompt delivery of functionalities.

Section 2 Background introduces the background of Smart City, data visualisation and the relationships between two topics. In this section, a literature review of related researches and representative works is also conducted along with experimenting existing tools and applications for visualisation of traffic data. The subsection 3.1 Data Characteristics describes the taxi operational data in detail.

Section 3 Project Specification includes the functionalities set to be delivered by Smart Taxi Vis, and a list of technologies used during this project. Section 4 Project Plan and Timetable includes a Gantt chart of this project. Section 5 Implementation describes the steps to pre-process the taxi dataset, implement visualisations and their interactive and coordinative functions. These sections were previously submitted as an initial project specification document [Wan17] by the author in May 2017.

Section 6 includes the testings conducted for Smart Taxi Vis and the feedback from testers, with documentation of Smart Taxi Vis in Section 7 Documentation. Observation and conclusion are drawn in Section 8 and Section 9 respectively. Section 10 describes the potential future improvement of Smart City Vis.

2 Background

The definition of Smart City has been adapting over the decades, the fundamentals of Smart City are to improve a city's efficiency sustainably [CDN11], thus improving the life quality of its citizens. To achieve sustainability means the optimisation of current resources is critical, by:

- Collecting operational data of the city via Information and Communication Technology (ICT) [DA11; Yue+09; AA08]
- Pre-processing the raw data into suitable formats [CGW15]
- Harvesting useful information via data analysis [CGW15; Pu+13; Zen+13]
- Policy-making base on the analytical results [LR12]

In order to carry out effective data analysis, the data collected need to be rigorously pre-processed and visualisations should be generated to convey the underlie information. This further requires appropriate data visualisation techniques.

2.1 Literature Review

This section reviews related literatures in the field of smart city and data visualisation.

2.1.1 Understanding Smart Cities: An Integrative Framework

Understand Smart Cities [Cho+11] is a comprehensive paper about smart city. The paper discusses the motivations and purposes behind smart city and gives a set of working definitions of a smart city, a city that monitors and connects its physical infrastructures with ICTs to leverage the efficiency, sustainability, equitability and liveability of the city.

The paper describes a framework with eight crucial factors of a successful smart city:

- Management and organisation
- Technology
- Governance
- Policy
- People and communities
- Economy
- Build infrastructure
- Natural environment

For each of the eight factors, authors detailed the challenges faced and the possible strategies to counter them. In particular, the paper emphasises that the inclusions of complex analytics to make better policies, requires visualisation techniques in the decision-making process.

Authors acknowledged that the capture of data from various ICT infrastructures is fundamental to a successful smart city, the decisions made upon those data acquired will offer huge potentials in optimising the operation of the city.

By gaining deep understanding of what a smart city is, this paper inspires the development of this project, precisely what the requirement specification of visualisation tool that suits the needs of a smart city.

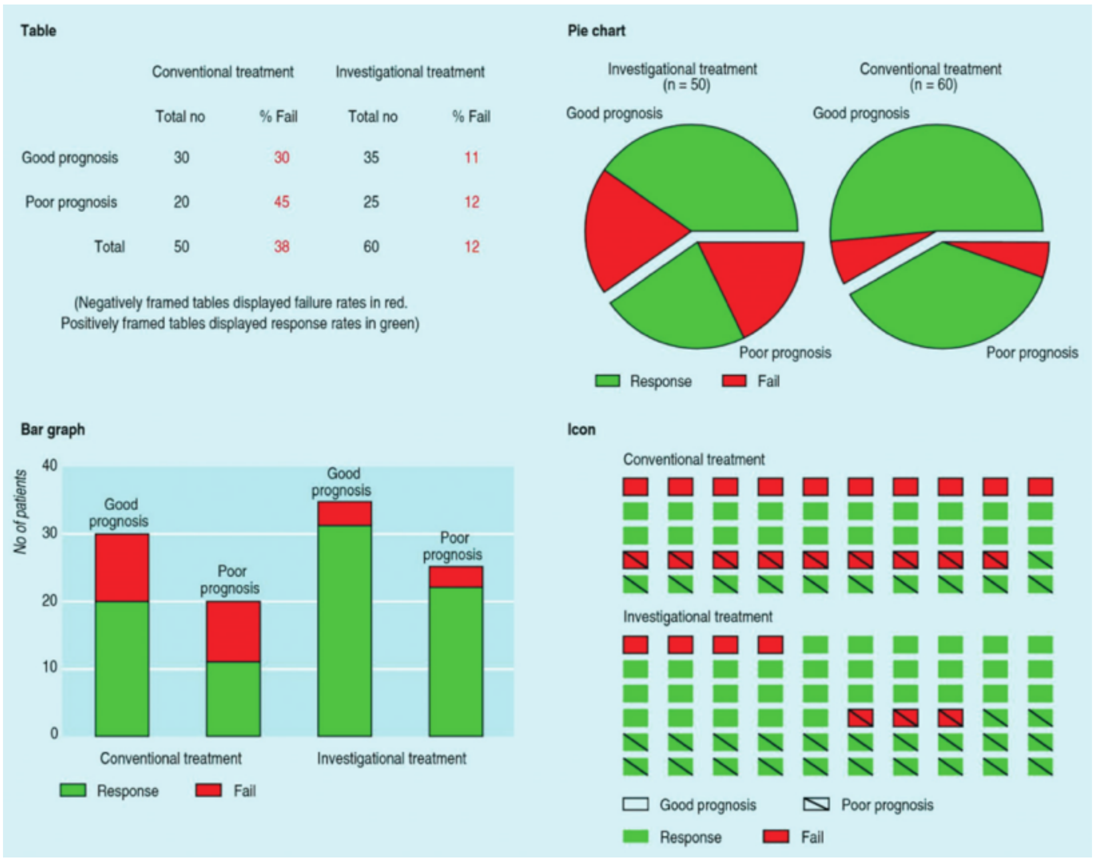
2.1.2 Interactive Data Visualization: Foundations, Techniques, and Applications

Interactive Data Visualization is a book [WGK10] introducing the fundamentals of data visualisation concepts and techniques. The book covers the spectrum of usage for data visualisation across different industries.

The book defines visualisation as “the communication of information using graphical representations”, the use of data visualisation techniques is prominent to convey more information underlie the dataset given via human visual cognition system. As mentioned by authors in the book, human as visual beings, absorbing information contained in graphs is more efficient than in text or in other formats [WGK10, pp. 3-5].

Data visualisation also plays an important role in the decision-making process. The same data presented in different visualisations will have different influence on the final decision. In Figure 1. is the study conducted by [Elt+99] on the influence of visualisation types on clinical decisions, it uses a table, a stacked bar graph, a pie chart and an icon display showing exactly the same dataset, the results of a clinical trial from both conventional and investigational treatments.

When a specially chosen group of competent physicians were asked to make a decision whether the clinical trial should be stopped or not, their decision accuracies were interestingly fluctuating from four visualisations, as seen in Table 1.



Table

	Conventional treatment		Investigational treatment	
	Total no	% Fail	Total no	% Fail
Good prognosis	30	30	35	11
Poor prognosis	20	45	25	12
Total	50	38	60	12

(Negatively framed tables displayed failure rates in red.
Positively framed tables displayed response rates in green)

Figure 1: Examples of the four types of display of hypothetical clinical trial data [Elt+99].

Visualisation Type	Decision Time	Decision Accuracy	Preference Rate
Table	35 seconds	68%	61.7%
Stacked bar graph	34 seconds	43%	23.5%
Pie chart	36 seconds	56%	14.8%
Icon display	37 seconds	82%	0%

Table 1: Performance on four types of visualisation in the study conducted [Elt+99].

Despite having the highest decision accuracy with similar time taken for making the decision, icon display was not preferred by any of the participants. Apart from the foundations of data visualisation, there are three chapters that are particularly valuable for this project:

- Visualization Techniques for Geospatial Data
- Visualization Techniques for Time-Oriented Data
- Visualization Techniques for Multivariate Data

In each chapter, not only that the authors presented the techniques for visualisations, but also listed down the issues faced during visualising the data. Since Smart Taxi Vis is mainly used for visualising those three types of data, this book is then extremely beneficial to the development.

2.1.2.1 Visualization Techniques for Geospatial Data

One of the planned visualizations will be to visualise the origin and destination of taxi dataset on a geographic flow map with interactive features. However, the points are often clustering together as shown in Figure 2, in order to generate a geographic flow map that effectively conveys meaningful message, in this chapter authors suggested that those points should be generalised. The process of generalisation involves simplifying points, which essentially means removing or combining the points that are not separately visible on a map with suitable scale [WGK10, pp. 247-249].

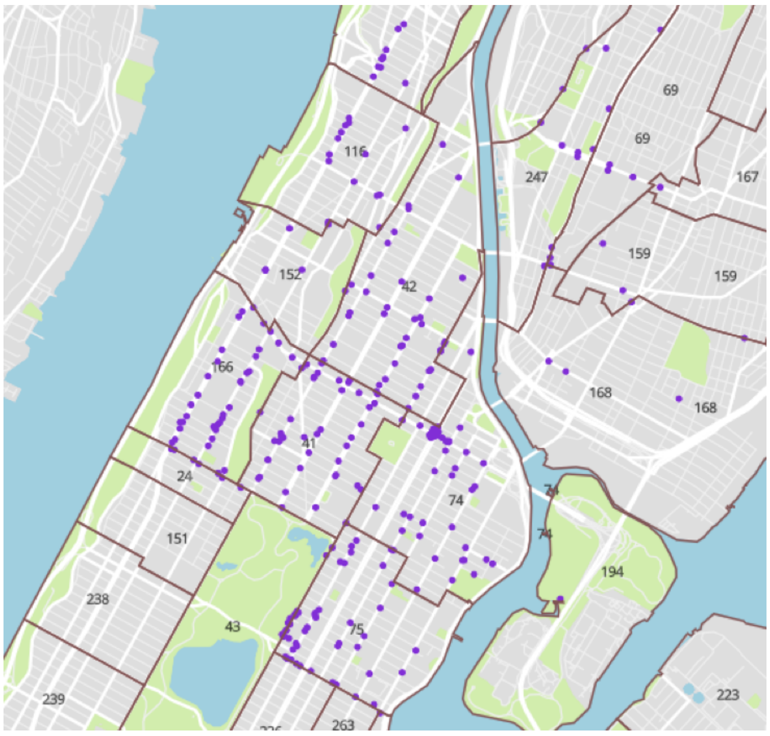


Figure 2: Visualisation of taxi pick-up locations in New York using Mapbox [Map17], as a prototype of Smart Taxi Vis.

2.1.2.2 Visualization Techniques for Time-Oriented Data

According to the authors, time itself is a dimension of any dataset with distinct characteristics, the importance of revealing patterns and relationships of time dimension with other dimensions is illustrated in this chapter. [WGK10, p. 254].

Figure 3 below shows the visualisations of one time-oriented dataset for cases of influenza occurred daily in Germany in a period of three years. The left visualisation is a simple line plot to linearly visualise the data, it clearly shows the peak times of influenza occurrence but the patterns underlie the dataset is hard to conclude. The centre (with one cycle represents 24 days) and the right (with one cycle represents 28 days) use a cyclic visualisation which utilises a spiral-shaped time axis.

There is no pattern to be recognised in the centre visualisation. By adjusting the cycle length to fit into the data's natural interval (multiple of 7 days), the right visualisation is obtained, which instantly reveals that more influenza cases occurred on Mondays than on any other days. This example demonstrates that the characteristics of time can significantly change "the expressiveness of visual representations" [WGK10, p. 254].

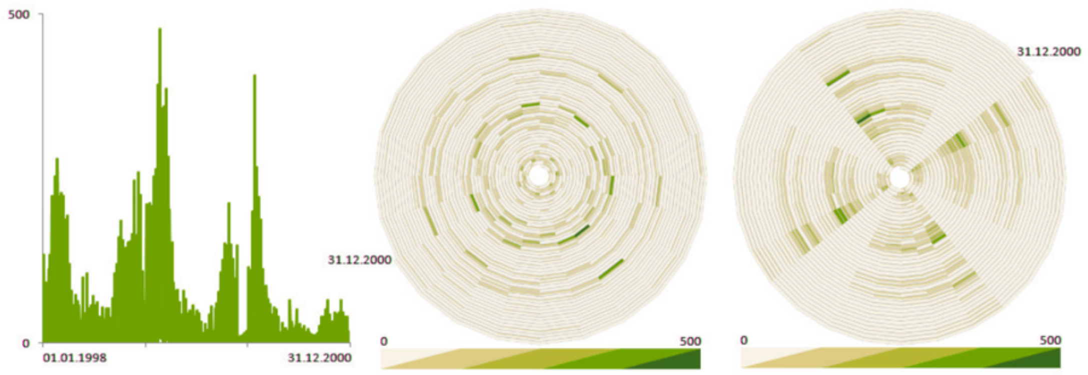


Figure 3: Linear vs cyclic visualisation of time-oriented data [WGK10, p. 255].

For the taxi dataset used for this project, discrete time data is introduced in multiple fields, e.g. taxi pickup time and dropoff time. According to the authors, discrete time data is the most widely used quantitative time model. Such dataset can be studied to find out the hidden patterns and 3D visualisation often provides a better illustration of time-oriented dataset [WGK10, pp. 258-263].

2.1.2.3 Visualization Techniques for Multivariate Data

In the field of data visualisation, multivariate data refers to datasets that have more than three variables. Visualisation of multivariate data is referred by Liu [Liu17] as ***Curse of Dimension*** due to the fact that the effectiveness of retinal visual elements such as colour, shape and size will deteriorate as the number of variables increases.

This chapter introduces visualisation techniques that maintain characteristics of multivariate dataset while providing effective renderings. The taxi dataset contains multiple point plots as the main data format, therefore the techniques for visualising point-based multivariate data are beneficial to the development of the proposed toolset. Four point-based techniques are described [WGK10, pp. 285-292] in this chapter as follows:

- **Dimension Subsetting** – dimensions can be selectively displayed by the user or the toolset automatically selects the most meaningful dimensions to visualise.
- **Dimension Reduction** – dimensionality reduction algorithms such as principal component analysis (PCA) will be applied to project higher dimensional dataset

into lower dimensions, meanwhile it tries to reduce the loss of information to the minimum during the process.

- **Dimension Embedding** – dimensions can be mapped to various graphical representations besides position, e.g. colour, size and shape. This is however limited as mentioned previously as ***Curse of Dimension***.
- **Multiple Displays** – visualisations are superimposed or juxtaposed. The classic example of multiple displays is the use of scatterplot matrix.

2.1.3 A Survey of Traffic Data Visualization

A Survey of Traffic Data Visualization [CGW15] is a paper that reviews representative works on the visualisation of large-scale, multi-modal and unstructured data captured by traffic systems. The paper describes conventional techniques of data processing and methods of visualising temporal, spatial, numerical, and categorical properties of traffic data, which provides constructive advices for feature proposal and planning of Smart Taxi Vis.

2.1.3.1 Pre-processing of Traffic Data

According to the authors, traffic datasets are usually high-dimensional and spatial-temporal, data pre-processing techniques are required before facilitating any useful understanding of the datasets. With a sequence of proper data pre-processing methods, traffic data visualisation can reveal traffic, social, geo-spatial and even economic patterns.

The authors separated data pre-processing into four steps:

- **Data Cleaning** - three steps in the data cleaning process are auditing data to find discrepancies, choosing transformations to fix discrepancies and applying the transformations to the entire dataset.
- **Data Matching** - match the recorded discrete data such as GPS coordinates on a map, fix any contradictory or inaccurate value.
- **Data Organisation** - the pre-processed data must be organised and stored in a database that supports interactive query of spatial-temporal data.
- **Data Aggregation** - reduce the data size over a large scale of space and time, by applying aggregation strategies for space, time (see Figure 4), movement direction and trajectory data.

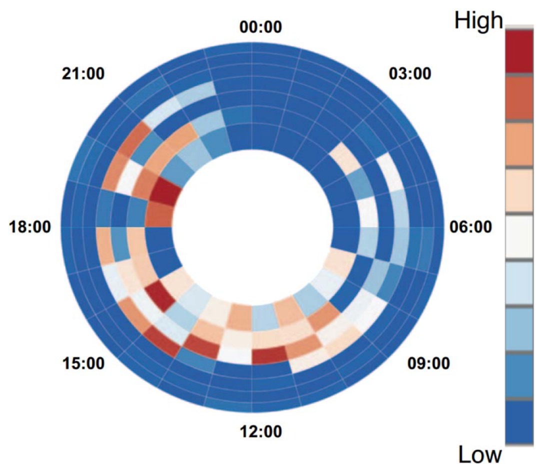


Figure 4: A radial layout that visualises the temporal data aggregated hourly, each ring represents a day [Pu+13]. Smart Taxi Vis adopts the same strategy to reduce data size.

2.1.3.2 Visualisation of Traffic Data

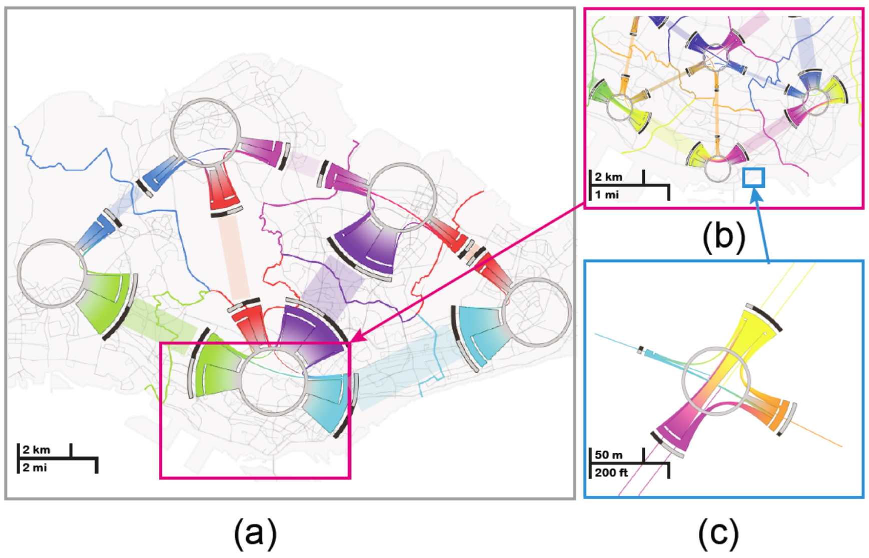


Figure 5: Region-based traffic data visualisation by Zeng et al., showing the traffic flow between (a) cities, (b) regions and (c) roads [Zen+13].

The authors suggested that the most important data dimensions of traffic data are time and space, relevant techniques are introduced in the paper and the following provide valuable advices to the development of Smart Taxi Vis.

- Visualisation of temporal dimension
  - **Linear time** - time as a linear field is the most widely used representation

of temporal dimension, it is capable of visualising the traffic data variation over time with a clear indication of peaks and valleys.

  - **Periodic time** - time as a periodic field, such as hours, weeks and days, often visualised by using a radial layout shown in 4, with the clear indication of patterns but low space efficiency.

- Visualisation of spatial dimension
  - **Point-based** - consider traffic data as discrete points, in combination with animation, straightforward observation of data patterns can be obtained. The advantage is that every object can be visualised individually but may cause overcrowding. Point-based approach is adopted in Smart Taxi Vis, see Figure 30.
  - **Region-based** - aggregate traffic data based on predetermined rules into regions such as streets, boroughs or other predefined boundaries. Comparing to point-based, region-based reveals patterns in a macro view rather than patterns of individual objects, Figure 5 is a region-based traffic data visualisation by Zeng et al. [Zen+13].

2.2 Similar Applications

This section reviews similar applications for traffic data visualisation.

2.2.1 MONiTOUR

MONiTOUR [Rat+16a] in Figure 6, is an e-waste tracker that visualises the travel routes of obsoleted printers, LCD and CRT monitors from the US. As a part of the joint project *e-trash Transparency Project*, MIT Senseable City Lab and Basel Action Network embedded GPS trackers into those e-wastes in order to obtain the travel routes.

The results are visualised on a world map, blue dots and red dots represent the origin and destination respectively, with white lines clearly depict that the majority of e-waste travelled to Asia. Filtering on device type and region. Figure 7 shows when a route is selected, the detailed travel history of the GPS tracker in steps.

The app was written in JavaScript with HTML 5 and the base map used was from Mapbox [Map17].

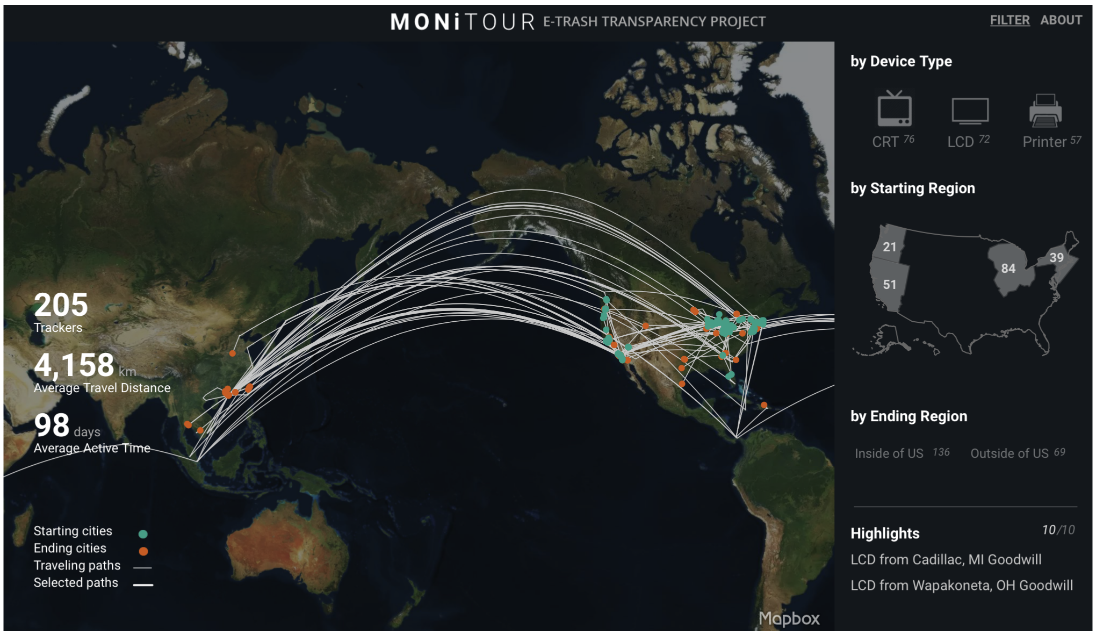
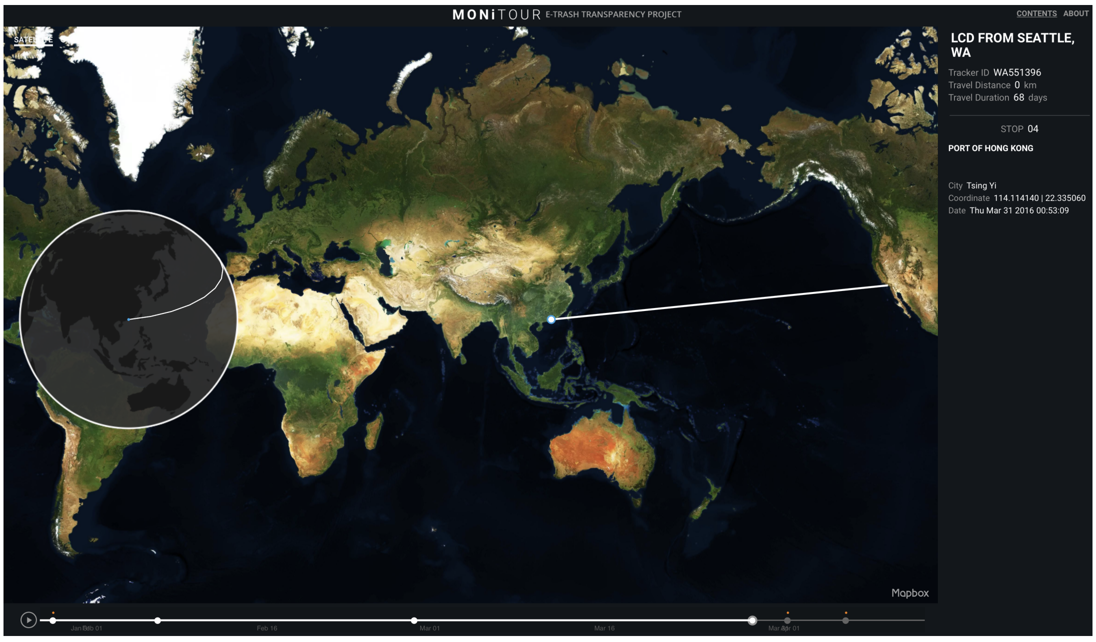


Figure 6: Interface of MONiTOUR visualising origins, destinations and paths of 205 GPS trackers embedded into printers, LCD and CRT monitors [Rat+16a].



LCD FROM SEATTLE, WA

Tracker ID WA551396
Travel Distance 0 km
Travel Duration 68 days

STOP 04

PORT OF HONG KONG

City Tsing Yi
Coordinate 114.114140 | 22.335060
Date Thu Mar 31 2016 00:53:09

Figure 7: Interface of MONiTOUR visualising the travel history of a GPS tracker embedded into a LCD monitor, from Seattle to Hong Kong [Rat+16b].

2.2.2 Visualizing MBTA Data

Visualizing MBTA Data is a web application developed by Barry and Card [BC14] for visualising subway data provided by Bostons Massachusetts Bay Transit Authority

(MBTA), for the month of February, 2014. It was written in JavaScript and HTML 5 with D3.js [Bos15]. It provides a set of powerful visualisations for three subway lines in Boston from different data dimensions subway trips, passenger counts and congestion time. Those three visualisations are inspirational to the development of Smart Taxi Vis.

2.2.2.1 Visualisation of Subway Trips

The subway trips are visualised using a Marey diagram, with a subway station map of Boston on the left of Figure 8. Each line on the Marey diagram indicates the path of one subway with X-axis being the entire trip, Y-axis being the time and vertical lines in light grey being subway stations along the subway line.

Steeper line represents a slower trip and denser lines indicate more frequent dispatches of subway. All lines can be highlighted by clicking and the corresponding dot that represents the subway on the map gets a halo effect on the map. Moving along the Y-axis of the Marey diagram updates dot positions on the map.

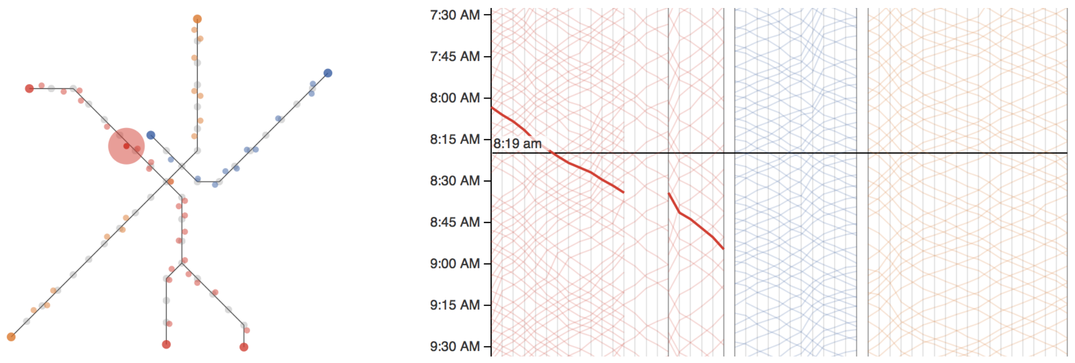


Figure 8: Visualizing MBTA Data: visualisation of subway trips with the lines and stations on the left map, the right is the visualisation of subway actual operating time on a Marey diagram [BC14].

2.2.2.2 Visualisation of Passenger Counts

Figure 9 visualises the entrances and exits per station. Each circle on the left map represents a station and the size depicts the busyness of the station in terms of number of passengers. The stations are ranked in a descending order with the heatmaps on the right visualise the data by hour and the histograms show the total number of passengers. Mouse hover the right section highlights the corresponding circle on the map.

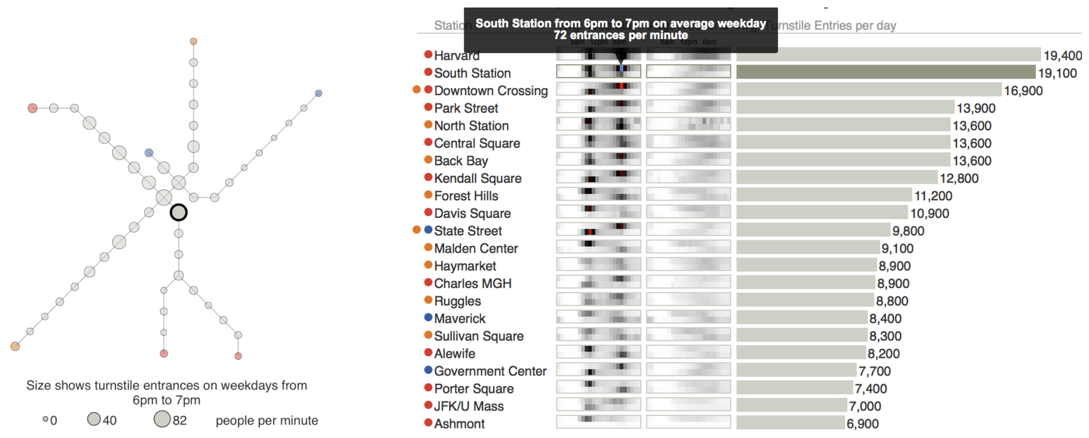


Figure 9: Visualizing MBTA Data: visualisation of passenger counts with the lines and stations on the left map, the right is the distribution of passenger on a heatmap and a histogram [BC14].

2.2.2.3 Visualisation of Congestion Time

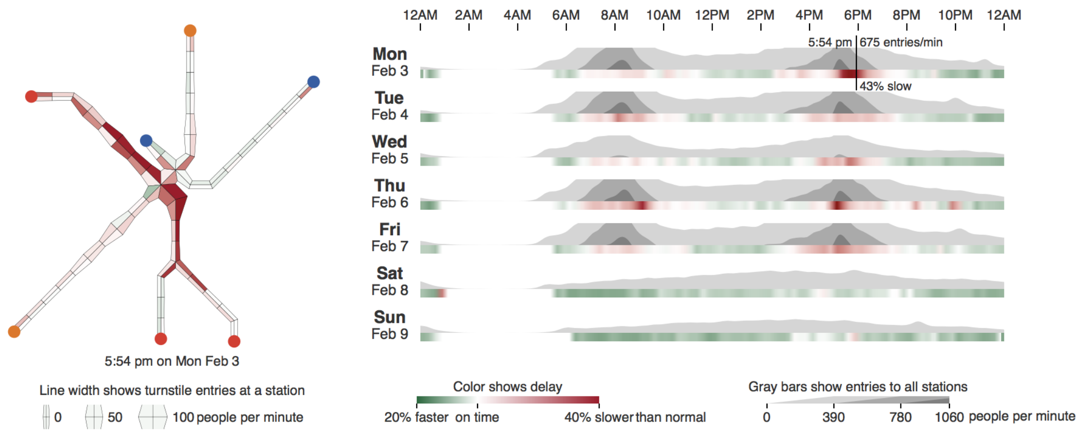


Figure 10: Visualizing MBTA Data: visualisation of congestion time with the lines and stations on the left map, the right is the number of passengers on a horizon chart in grey above the congestion time on a heatmap [BC14].

Figure 10 visualises one week of congestion time data of Boston subway lines and its relationship with passenger count. Each glyph on the left map represents a station with the thickness indicating the number of passengers and the colour indicating delays in minutes from both directions. The peaks of horizon chart on the right coincide with the heatmap below by large, which implicitly suggests their relationships.

2.2.3 Boston 311

Boston 311 [GLO12] in Figure 11 is a geospatial visualisation of 3-1-1 incident reports in Boston from October 2010 to June 2012.

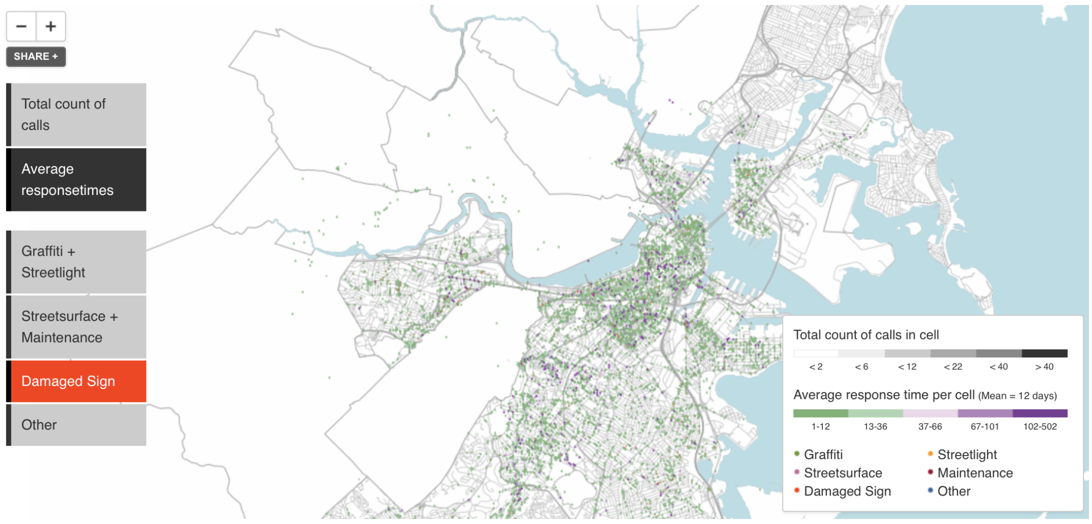


Figure 11: Interface of Boston311 visualising average response time against damaged sign in Boston [GLO12].

Two dimensions can be visualised simultaneously by selecting them in the left panel. The first dimension can be either total count of calls or average response time, whereas the second dimension can be chosen from graffiti and streetlight, streetsurface and maintenance, damaged sign or other.

2.3 Limitation of Similar Applications

This section analyses the limitation of similar applications reviewed in the previous section.

2.3.1 MONiTOUR

The performance of MONiTOUR is a major concern for visualisation of millions of taxi operational data entries. During the experiment of MONiTOUR, even there are only 205 distinct paths visualised on the map, the performance is not optimal. The web app lags significantly when applying zoom and pan operations, using Chrome (Version 58.0.3029), Firefox (Version 52.1.0) and Safari (Version 10.1). The initialisation of the application also takes more than ten seconds with a fast internet connection.

Certain level of filtering is allowed in MONiTOUR but is far from sufficient for visualisation of taxi operational data, due to the high data dimensionality and huge data size.

Visualisation of time-oriented data is absent from the application.

2.3.2 Visualizing MBTA Data

Visualizing MBTA Data is a great application for visualising subway traffic data, it successfully reveals hidden patterns and insights buried within the dataset with swift performance. No limitation was discovered by the author, however the professionalism might be a burden for average users.

2.3.3 Boston 311

Boston 311 successfully visualises high volume of data with satisfactory performance. However, a generalisation process mentioned in section *Visualization Techniques for Time-Oriented Data* is missing and causes data points to be clustered and overlapping each other regardless of zoom level. This creates confusions during presentation stage for the users.

Filtering is extremely limited in Boston 311, apart from selecting different dataset, users are unable to select subset of data for visualisation.

Visualisation of time-oriented data is absent from this application.

3 Project Specification

The primary objective of this project is to develop a visualisation toolset call Smart Taxi Vis, for visualising the taxi operational data, providing extra functionalities based on the existing JavaScript libraries. The toolset should contain the following major features:

- Converting large chunks of taxi operational data from CSV format to JSON.
- Exploring the data via various types of visualisation.
- Analysing the visualisation via different interactive functions such as filtering of data dimension or volume.

The project would help the author to gain a deeper understanding of information visualisation and geospatial visualisation, the usefulness and the difficulties in obtaining such visualisation. Due to the massive amount of data used, data pre-processing will also be focused during the development phase.

3.1 Data Characteristics

3.1.1 Taxi Operational Data recorded by the New York City Taxi & Limousine Commission (TLC)

Taxicab & Livery Passenger Enhancement Programs (TPEP/LPEP) was introduced in March 2004 by the TLC, the primary objective of this program is to improve New

York taxi service by implementing specific technology to collect taxi operational data, including pick-up and drop-off location, trip distance and time, passenger counts and others. Prior to the TPEP/LPEP, this data collection process was manually done by the drivers using a hand-written log book, the introduction of the program significantly improved accuracy of the data and efficiency of the data collection process [NYC].

Table 3 shows the description of the data provided by TLC at `http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml`.

For each month, three datasets in CSV format are provided, Yellow Taxi, Green Taxi and For-hire Vehicle (FHV). Due to the nature of FHV service, its dataset contains only pick-up time and pick-up borough ID, thus the design of Smart Taxi Vis did not take the FHV dataset into consideration.

Yellow Taxi is the traditional taxi operating in New York, however TLC found that 95% of taxi trips occurred in Manhattan Central Business District[1] and airports, resulting in taxi shortage in outer boroughs of New York. Green Taxi was therefore introduced in 2013 as the Boro Taxi Program, permits Green Taxis to pick up passengers in only under-served boroughs, to fill in the gap [Com13]. Smart Taxi Vis utilities Green Taxi data collected for the week starting from 5th Dec 2016 to 11th Dec 2016, which contains 265,222 records in total.

Table 2 shows the difference between three datasets in terms of dataset size and number of records.

Dataset	**Dataset size**	**Number of records**
Yellow Taxi	1800 MB	11,500,000
Green Taxi	200 MB	1,500,000
For-Hire Vehicle (FHV)	300 MB	8,500,000

Table 2: Taxi operational datasets [NYC17]. Each month the size and number of records vary slightly, all data shown in the table are the estimated average.

Column Name	**Data Type**	**Data Description**	**Sample**
VendorID	Integer	Vendor that provided the data 1= Creative Mobile Technologies, LLC 2= VeriFone Inc.	1
Lpep_pickup_datetime	Datetime	Taxi pick-up date time	01/01/2015 02:21
Lpep_dropoff_datetime	Datetime	Taxi drop-off date time	01/01/2015 03:21

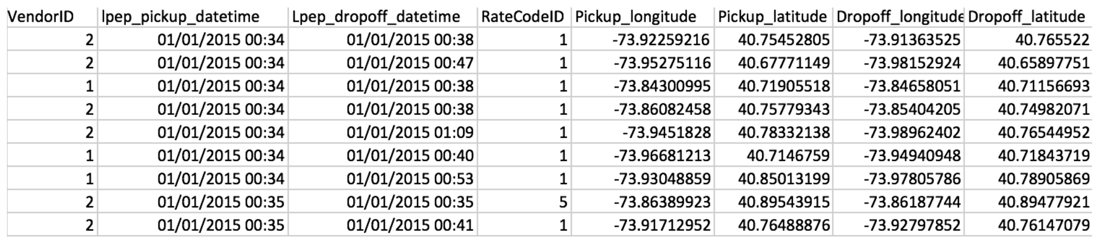
[1]Midtown Manhattan and Lower Manhattan

RateCodeID	Integer	Rate code for the trip 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare 6=Group ride	1
Pickup_longitude	Longitude	Taxi pick-up longitude	73.86389923
Pickup_latitude	Latitude	Taxi pick-up latitude	40.72250748
Dropoff_longitude	Longitude	Taxi drop-off longitude	73.94389923
Dropoff_latitude	Latitude	Taxi drop-off latitude	40.70850748
Passenger_count	Integer	Number of passengers for the trip	5
Trip_distance	Double	Total distance for the trip	14.01
Fare_amount	Double	Total fare for the trip	16.55
Extra	Double	Extra surcharge for the trip	0.5
MTA_tax	Double	Metropolitan Transit Authority Tax for the trip	0.5
Tip_amount	Double	Total tip amount for the trip	3.5
Tolls_amount	Double	Total toll amount for the trip	5.33
Improvement_surcharge	Double	Taxi Improvement Fund surcharge	0.3
Total_amount	Double	Total charge amount for the trip (Fare + all surcharges)	20.55
Payment_type	Integer	Payment type used for the trip 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip	1

Table 3: Description of taxi operational data collected by TLC [NYC15].

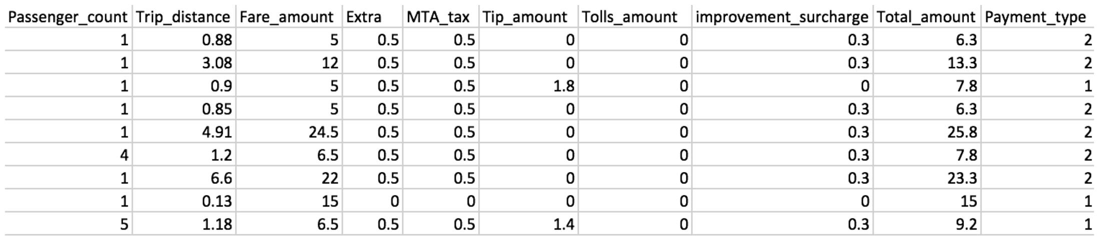
Starting from July 2016, instead of the coordinates, only pick-up and drop-off borough ID are provided, as the result only data prior to that are used in this project. Figure 12 and Figure 13 are screenshots of Green taxi operational dataset from January 2015.

The dataset contains both geospatial (longitude and latitude) and discrete time-oriented data (pick-up and drop-off time).



VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude
2	01/01/2015 00:34	01/01/2015 00:38	1	-73.92259216	40.75452805	-73.91363525	40.765522
2	01/01/2015 00:34	01/01/2015 00:47	1	-73.95275116	40.67771149	-73.98152924	40.65897751
1	01/01/2015 00:34	01/01/2015 00:38	1	-73.84300995	40.71905518	-73.84658051	40.71156693
2	01/01/2015 00:34	01/01/2015 00:38	1	-73.86082458	40.75779343	-73.85404205	40.74982071
2	01/01/2015 00:34	01/01/2015 01:09	1	-73.9451828	40.78332138	-73.98962402	40.76544952
1	01/01/2015 00:34	01/01/2015 00:40	1	-73.96681213	40.7146759	-73.94940948	40.71843719
1	01/01/2015 00:34	01/01/2015 00:53	1	-73.93048859	40.85013199	-73.97805786	40.78905869
2	01/01/2015 00:35	01/01/2015 00:35	5	-73.86389923	40.89543915	-73.86187744	40.89477921
2	01/01/2015 00:35	01/01/2015 00:41	1	-73.91712952	40.76488876	-73.92797852	40.76147079

Figure 12: Partial screenshot of Green Taxi operational dataset from January 2015 [NYC17].



Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	improvement_surcharge	Total_amount	Payment_type
1	0.88	5	0.5	0.5	0	0	0.3	6.3	2
1	3.08	12	0.5	0.5	0	0	0.3	13.3	2
1	0.9	5	0.5	0.5	1.8	0	0	7.8	1
1	0.85	5	0.5	0.5	0	0	0.3	6.3	2
1	4.91	24.5	0.5	0.5	0	0	0.3	25.8	2
4	1.2	6.5	0.5	0.5	0	0	0.3	7.8	2
1	6.6	22	0.5	0.5	0	0	0.3	23.3	2
1	0.13	15	0	0	0	0	0	15	1
5	1.18	6.5	0.5	0.5	1.4	0	0.3	9.2	1

Figure 13: Partial screenshot of Green Taxi operational dataset from January 2015 [NYC17].

A .shp file containing boundaries of 354 taxi zones provided by TLC can be downloaded from `https://s3.amazonaws.com/nyc-tlc/misc/taxi_zones.zip`. .shp format is a popular geospatial vector data format for geographic information system (GIS) software.

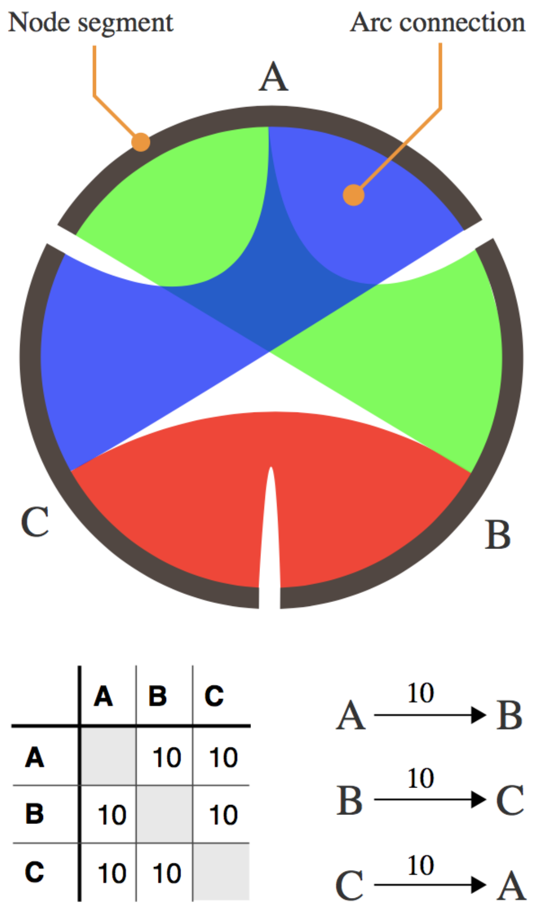
3.2 Feature Specification

Smart Taxi Vis is an interactive and responsive web application that contains the following features:

1. A Map View that predominantly serves as a geospatial visualisation but also contains other information dimensions.
2. A Chord Diagram for the visualisation of inter-relationships between taxi zones.
3. A Histogram visualises the distribution of taxi data by geospatial dimension or time-oriented dimension.
4. Interactions and coordinations between three types of visualisation.
5. User options to adjust time in order to present time-oriented data dimension..
6. User options to filter the selection of data to be visualised.

3.2.1 Chord Diagram

Chord diagram visualises the inter-relationships between entities [Rib13]. A node segment or a chord represents one entity and an arc connection or a path connects two entities. The data is in a square matrix form, which maps the relationship and the magnitude between entities. Below in Figure 14 is an illustration of a chord diagram with its dataset.



	A	B	C
A		10	10
B	10		10
C	10	10	

Figure 14: A sample chord ciagram and its dataset [Rib13].

Features proposed:

1. A chord diagram for taxi data visualisation.
2. A colourmap that serves as the legend.
3. User options to choose which data dimension to be visualised.
4. Filter options to choose the amount of data to be visualised.
5. Filter options to choose the time dimension to be visualised.
6. Both chord and path responses to mouse hover and mouse click event.

3.2.2 Map View

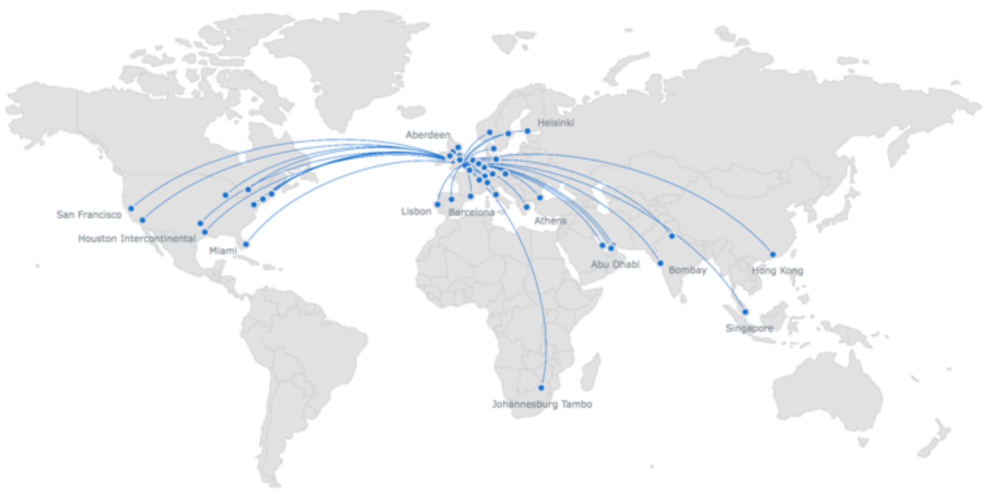


Figure 15: An example of a connector map using AnyChart.js [Any17].

Features proposed:

1. A base map visualising all the taxi zone boundaries.
2. A dot map visualising the selected taxi zones, with colours map to its data dimension.
3. A connector to connect taxi zones upon clicking on the chord diagram.
4. A choroplethly highlighted taxi zone upon clicking on the Histogram.
5. The map responses to mouse hover and mouse click event.

3.2.3 Histogram View

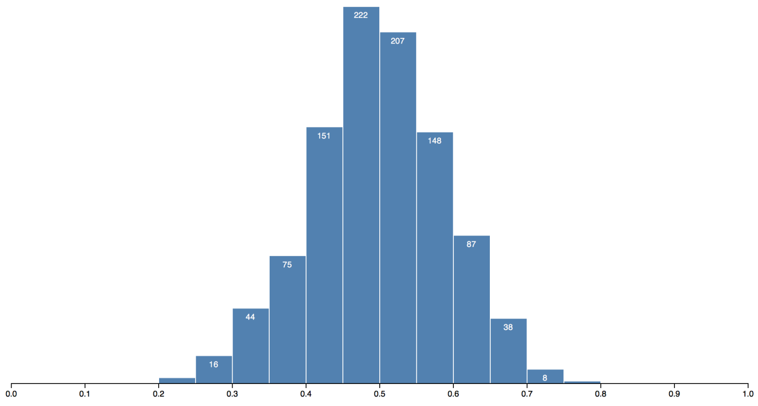


Figure 16: An example of ahistogram using D3.js [Bos16].

Features proposed:

1. A histogram visualising the distribution of the data dimension.
2. User options to choose the distribution dimension being visualised, either zone based or time based.
3. A connector to connect taxi zones upon clicking on the chord diagram.
4. A choroplethly highlighted taxi zone upon clicking on the Histogram.
5. The histogram responses to mouse hover and mouse click event.

3.3 Technology Choices

This section introduces the fundamental technologies used, different technologies will be used during different phase of the development, to suite the needs during that phase.

The primary hardware for the development will be a MacBook Pro embedded with an Intel i7-6920HQ CPU and 16GB of RAM, with the operating system of macOS Sierra 10.12.4. This will ensure that the large dataset can be pre-processed efficiently and the hardware is sufficient to run virtual machines of other operating system for testing purpose.

3.3.1 Data Pre-processing

Java is chosen as the primary language for data pre-processing. Java is one of the most popular programming languages that has great cross platform support with detailed documentation. The author also has years of experience in Java, thus Java is chosen to expedite the data pre-processing phase.

GSON [Goo08] is a Java library that enables Java for manipulating JSON format in accordance with JSON specification RFC4627 [Cro06]. Taxi operational data in CSV are piped into a MySQL database and then retrieved using a Java program written.

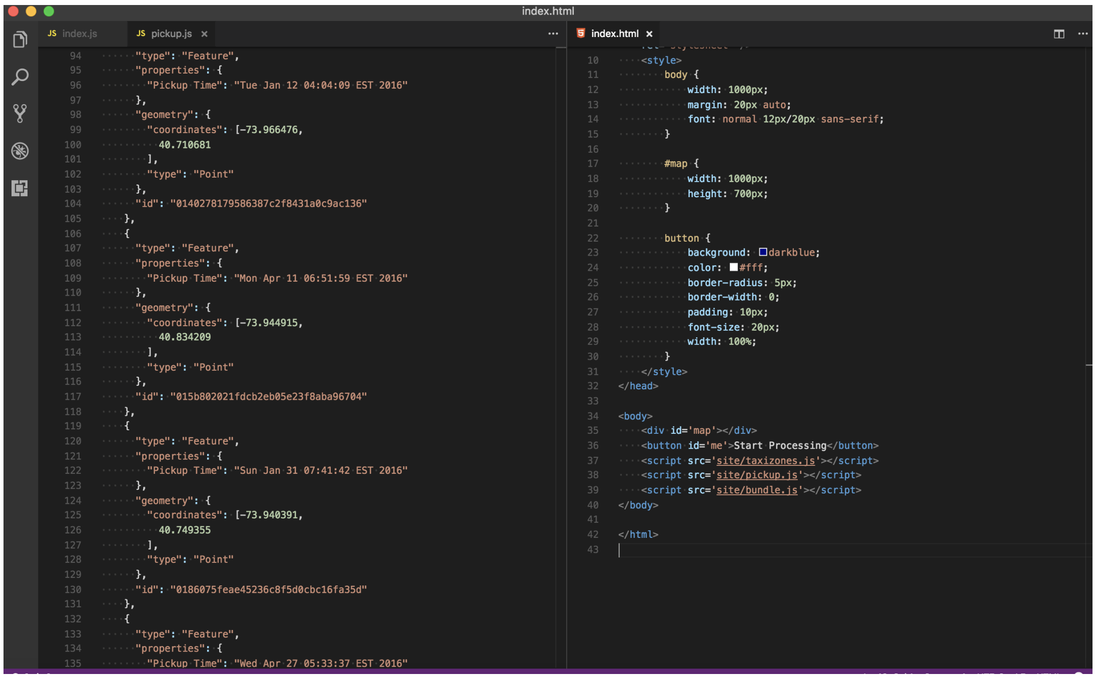
The IDE for Java will be IntelliJ IDEA Community version 2017.1.5 with JDK build 1.8.0_121b1.

3.3.2 Web Application Development

Smart Taxi Vis will be presented in the form of web application, which means all interactive functions will be manipulated by users via browsers. The application will mainly use HTML and CSS for presentation and JavaScript for functionality.

This combination is supported by any modern device with a screen, there are many existing frameworks such as ASP.NET MVC [Mic15], Angular [Goo17] and Bootstrap [Boo16] to expedite the development process.

Microsoft Visual Studio IDE [Mic17] was considered to be the IDE for the development of web Application. The IDE is fully-featured with development, testing and publication of the application. However, it only supports Windows and the publication



```
      "type": "Feature",
      "properties": {
        "Pickup Time": "Tue Jan 12 04:04:09 EST 2016"
      },
      "geometry": {
        "coordinates": [-73.966476,
          40.710681
        ],
        "type": "Point"
      },
      "id": "0140278179586387c2f8431a0c9ac136"
    },
    {
      "type": "Feature",
      "properties": {
        "Pickup Time": "Mon Apr 11 06:51:59 EST 2016"
      },
      "geometry": {
        "coordinates": [-73.944915,
          40.834209
        ],
        "type": "Point"
      },
      "id": "015b802021fdcb2eb05e23f8aba96704"
    },
    {
      "type": "Feature",
      "properties": {
        "Pickup Time": "Sun Jan 31 07:41:42 EST 2016"
      },
      "geometry": {
        "coordinates": [-73.940391,
          40.749355
        ],
        "type": "Point"
      },
      "id": "0186075feae45236c8f5d0cbc16fa35d"
    },
    {
      "type": "Feature",
      "properties": {
        "Pickup Time": "Wed Apr 27 05:33:37 EST 2016"
```

```
    <style>
        body {
            width: 1000px;
            margin: 20px auto;
            font: normal 12px/20px sans-serif;
        }

        #map {
            width: 1000px;
            height: 700px;
        }

        button {
            background: darkblue;
            color: #fff;
            border-radius: 5px;
            border-width: 0;
            padding: 10px;
            font-size: 20px;
            width: 100%;
        }
    </style>
</head>

<body>
    <div id='map'></div>
    <button id='me'>Start Processing</button>
    <script src='site/taxizones.js'></script>
    <script src='site/pickup.js'></script>
    <script src='site/bundle.js'></script>
</body>

</html>
```

Figure 17: Interface of Visual Studio Code editing HTML and JavaScript files.

and hosting of the application is tied to Windows Azure. Therefore, Microsoft Visual Studio Code [Mic16], the code editor that inherits a certain level of functionality from the IDE, will be used for the development. For the same reason, ASP.NET MVC framework was abandoned after a short period of development.

Visual Studio Code is an open source (MIT License) code editor and has a huge library of extensions with active communities, it supports all major operating system. More importantly, it is capable of handling all the tasks involved in the development.

Visual Studio Code version 1.11.2 will be used for completing the web application.

3.3.3 Testing and Documentation

Testing and documentation are two essential steps for software development. According to Vliet[Vli07, p. 386], the cost of repairing errors made at an early stage is extremely high if they are discovered at a late stage of the development, thus the testing should be frequently conducted during the development to minimise the negative impact of errors and bugs.

A set of tests will be conducted against the feature specification with the following tests being focused:

- Unit Testing
- Integration Testing
- Functional Testing
- Usability Testing
- Compatibility Testing

JSDoc [JSD14] is a Doxygen-like [Dox16] documentation generating tool specifically designed for JavaScript. It is free under the Creative Commons Attribution-ShareAlike 3.0 Unported License. JSDoc is capable of generating well structured documents base on the comments in the code. It will also produce class hierarchy and collaboration diagrams. A HTML version will also be generated, makes it easier to publish the documentation online.

Using JSDoc also encourages the author to comment the code properly during development, improves code quality and reduces bugs.

3.3.4 Others

3.3.4.1 Version Control

Git [Sof17] is the most widely used version control system that is free (GNU General Public License v2) and supports all major operating systems.

The free Git repository hosting service GitHub will be used to store all the code. GitHub will also serve as a regular backup warehouse to prevent any accident that may result in lost or corrupted files. Some interesting insights about Smart Taxi Vis's development can be observed in Appendix 13.2, or visit the project repository at `https://github.com/WangQiru/SmartTaxiVis`.

Git version 2.11.0 with the command line tools will be used throughout the entire development lifecycle.

3.3.4.2 Project Management

Zube [Piv17] is an Agile project management web application that integrates with GitHub, it will be used to track the development progress of Smart Taxi Vis. The advantage of utilising such tool is that it provides a clear view of pending development as well as issues discovered.

Since it integrates with GitHub, issue management is simplified. New sub-task in an agile sprint created on Zube gets synchronised to GitHub as issues and can be completed via git commit keyword or on Zube's web application interface.

```
git commit -m 'Fix #15'   or   git commit -m 'Resolve #15'
```

Source Code 1: git command to add changes made with message 'Fix #15' or 'Resolve #15', both ***fix*** and ***resolve*** are keywords that will trigger GitHub to close issue 15.

3.3.4.3 Code Quality Assurance

Multiple measures are taken to ensure the code written fulfils the standard of Department of Computer Science and minimise the number of bugs.

Following ten rules from ***Bob's Concise Coding Conventions*** [Lar09] is the primary measure taken to avoid producing illegible code. The rules are concise and easy to follow, with effectivity in improving reusability and legibility of the code.

Linting [Joh78] is a process of statically checking bugs and obscurities and enforcing a set of rules and restrictions in the code. Elint [Esl17] as the most popular open source linter for JavaScript is used during development.

In combination with the adoption of a popular set of linting rules generated from Airbnb's JavaScript Coding Style Guide [Air17], common mistakes will be detected in real time during development, this improves the quality of code produced and dramatically reduces bugs.

4 Project Plan and Timetable

4.1 Development Strategy

There are five essential stages of software development [Lar11; Vli07, p .11].

Five sections that are included in the gantt chart:

1. Requirements specification
2. Software design
3. Implementation
4. Testing
5. Documentation

Arnuphaptrairong [Arn11] discovered that project planning is the area where the most number of risks originated. Therefore, Smart Taxi Vis will strictly adopt the Scrum Agile software development framework to plan and control the project development throughout the entire lifecycle.

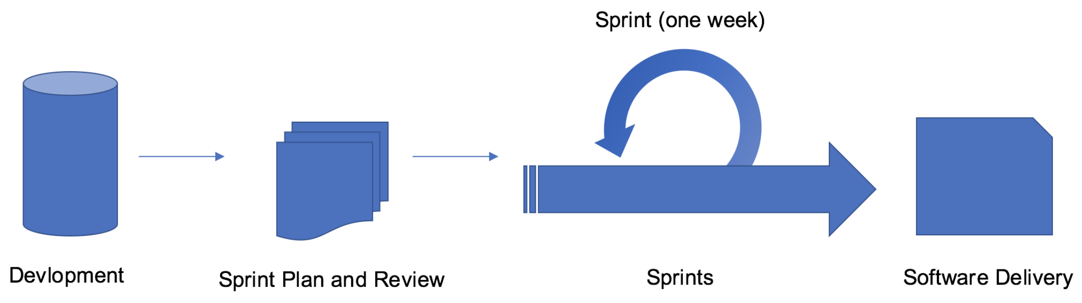


Figure 18: Slightly modified Scrum workflow for Smart Taxi Vis.

Scrum is the most widely adopted Agile framework. Figure 18 shows a Scrum workflow tailored for Smart Taxi Vis. Development is broken into smaller Sprints, each Sprint takes one week to develop, completion of all Sprints signals the shipment of software. The original Scrum workflow requires a daily team meeting during each Spring, since the author is the sole developer in the team, a daily meeting with team members during each Spring is cancelled. Sprint requires planning ahead and reviewing after, at the end of each Sprint, the development team will review it with stakeholders to identify improvements or corrections needed.

This fits exactly into the weekly supervision meeting where new features are discussed and completed features are reviewed. The flexibility of Scrum allows the author to embrace changes to requirements and correct misinterpretation of requirements within an acceptable timeframe.

4.2 Gantt Chart

Figure 19 shows the gantt chart for Smart Taxi Vis.

4.3 Risk Analysis

This analysis outlines the potential risks that will have catastrophic damage on the development or severely affect the progress of development and strategies to counter them.

4.3.1 Risk Identification

The identification of risks followed the research carried out by Arnuphaptrairong [Arn11]. Seven risks are rated as the most frequently encountered in software development lifecycle in the research as follows:

1. Misunderstanding of requirements
2. Lack of top management commitment and support
3. Lack of adequate user involvement
4. Failure to gain user commitment
5. Failure to manage end user expectation
6. Changes to requirements
7. Lack of an effective project management methodology

4.3.2 Risk Mitigation

This section outlines the mitigation measures against the risks identified.

Table 4 shows the risk identified by the author. The first risk, the involvement of new technology with a high level of technical complexity is rated as the riskiest factor. Obstacles are expected in the project due to the novelty of the author, the involved technology will be carefully selected by the following factors:

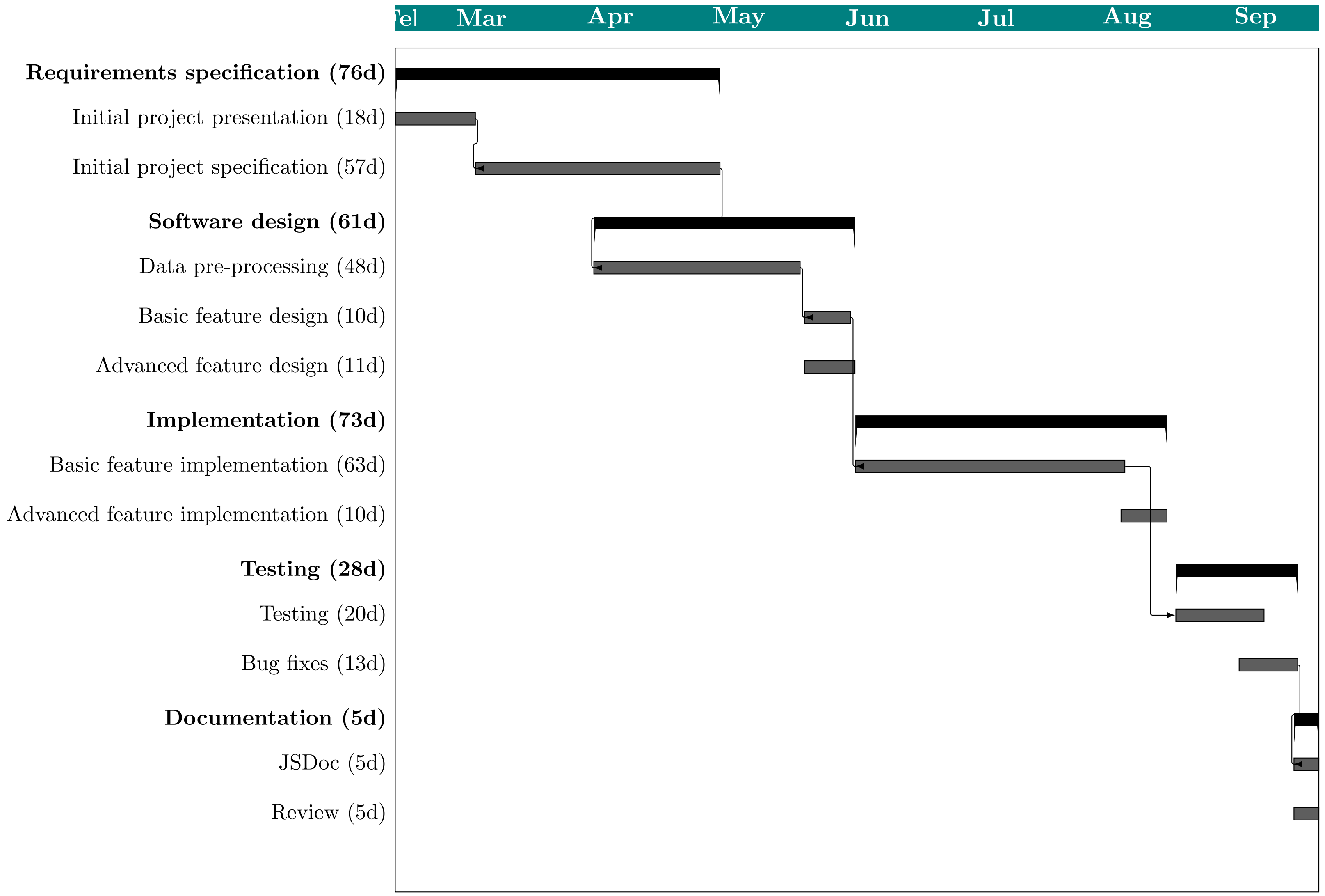


Figure 19: Smart Taxi Vis project gantt chart.

- Language structure and syntax are close to the author's prior experience
- Detailed documentation for the technology
- Preferably free and open source
- Active communities are in place
- Successful software built upon the technology can be accessed and reviewed

The second risk, misinterpretation of requirements has serious consequence, it mitigated by conducting regular meeting with supervisor. Scrum workflow also helps minimise the risk.

The third risk, Java and JavaScript are the main languages used for the development of Smart Taxi Vis, they are both well-documented and have active communities to provide technical support. This mitigates the risk from inadequate programming skills of the author.

The fourth risk, the development will start with high-value features with low-risk, proceeding to low-value features with low-risk and eliminating or substituting low-value features with high-risk. This ensures that the final project will deliver useable features.

The fifth risk, two volunteers with data visualisation background are chosen to test the features at the end of each or several Sprints, provide insightful feedback to minimise the risk brought by insufficient user involvement and feedback.

The sixth risk, GitHub, Dropbox and a local external hard drive are used for backup to ensure any equipment failure will not result in delay of the development.

The seventh risk, lack of effective project management skill, is mitigated by adopting Scrum Agile software development framework mentioned in Section 4.1 Development Strategy.

The last risk is personal illness which is unavoidable, ensuring GP Surgeries are within reach can help minimise the damage.

	Risk	Probability	Impact	Mitigation measures
1	Involvement of new technology with a high level of technical complexity	High	High	Choose the technology involved carefully, avoid unrealistic or unnecessary features and invest more time into exploring the technology used
2	Misinterpretation of requirements	Medium	High	Regular meeting with supervisor for consultations and progress monitoring
3	Inadequate skills and knowledge in programming	Medium	High	Experienced programming language with detailed documentation and active community is used
4	The final software developed has serious bugs that makes it unusable	Medium	High	Features are developed in an order ranked by their value/risk ratio
5	Insufficient user involvement and feedback	Medium	Medium	Volunteers with data visualisation background are regularly surveyed when each version rolls out
6	Equipment failure resulting in loss of files	Low	High	GitHub and Dropbox are used for regular backup of files. Laptop is backup daily using an external harddrive
7	Lack of effective project management skill	Low	High	Adopt Scrum Agile software development framework strictly
8	Personal illness	Low	Medium	GP Surgeries are within reach

Table 4: Risk analysis and mitigation.

5 Implementation

Smart Taxi Vis consists of a main page (Figure 20) with visualisations and a documentation site. The main page consists of user options to toggle animation of Chord Diagram, switch colour dimensions and switch histogram dimensions. Two sliders are for the filtering of taxi zones and selecting hour of the day.

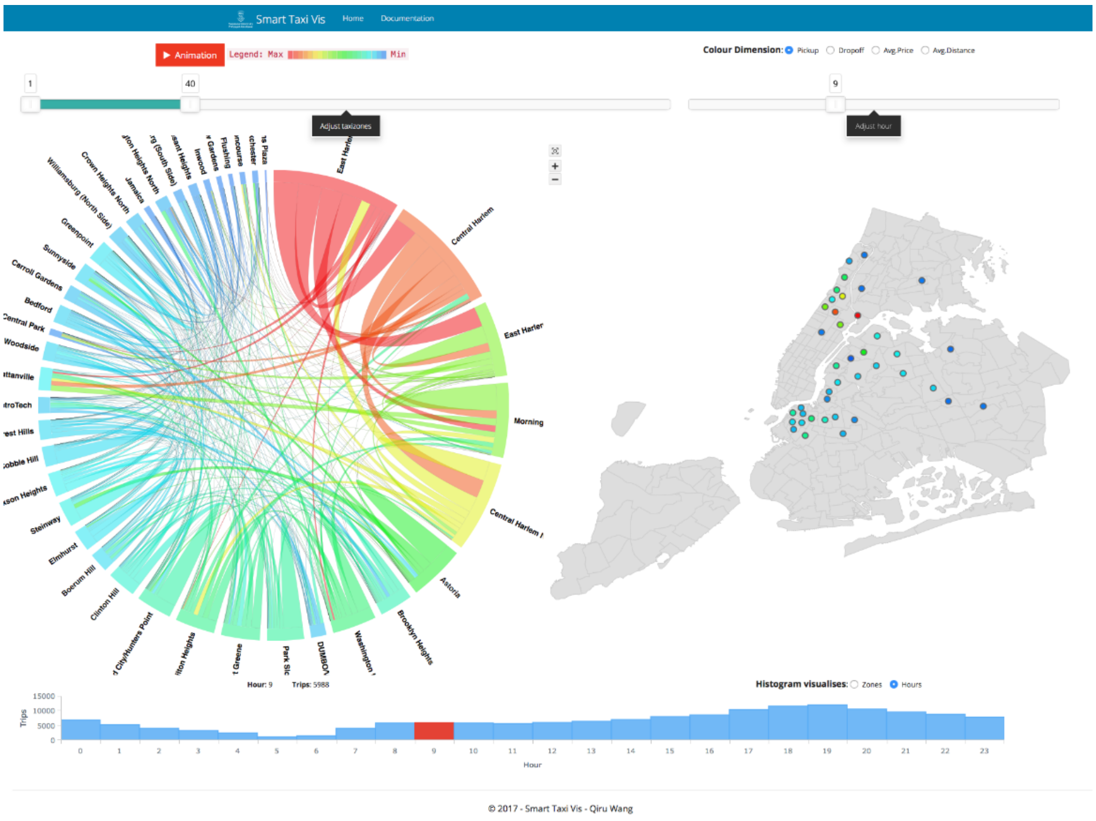

Smart Taxi Vis
Home
Documentation
Animation
Legend: Max
Min
Colour Dimension: Pickup Dropoff Avg.Price Avg.Distance
Adjust taxizones
Adjust hour
Hour: 9
Trips: 5988
Histogram visualises: Zones Hours
Trips
Hour
© 2017 - Smart Taxi Vis - Qiru Wang

Figure 20: The main page of Smart Taxi Vis

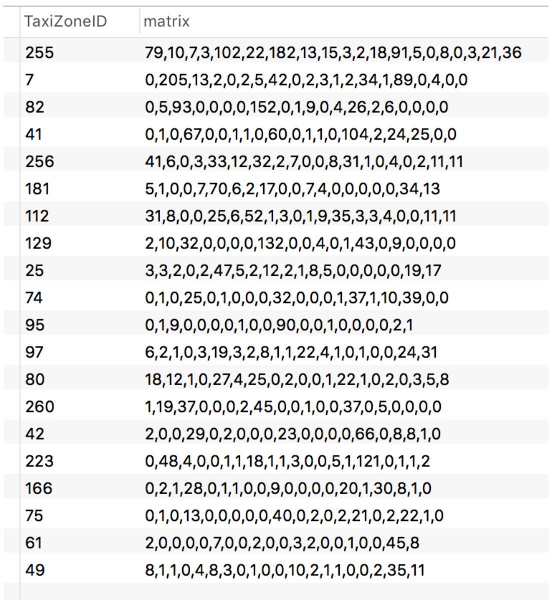
The rest of this section describes the different stages of the implementation in detail.

5.1 Data Pre-processing

A subset (one week) of the data was imported into a MySQL database using MySQL workbench [Ora15] before the data pre-processing steps below:

1. Filtering out columns that are not used
2. Filtering out records with errors
3. Deriving new columns

The use of MySQL database simplifies these steps via customised SQL query statements. In queries, predicates are used for filtering out unwanted data records. Combining all queries into Stored Procedures will significantly improve the performance. Instead of retrieving all the data and manipulate them locally, Stored Procedures serve as APIs of the database, filter out the unnecessary data and only return those that are requested in the desired format. In Figure 21, column matrix is a square matrix and can be used to render a chord diagram directly by putting them into a JavaScript array object.



TaxiZoneID	matrix
255	79,10,7,3,102,22,182,13,15,3,2,18,91,5,0,8,0,3,21,36
7	0,205,13,2,0,2,5,42,0,2,3,1,2,34,1,89,0,4,0,0
82	0,5,93,0,0,0,0,152,0,1,9,0,4,26,2,6,0,0,0,0
41	0,1,0,67,0,0,1,1,0,60,0,1,1,0,104,2,24,25,0,0
256	41,6,0,3,33,12,32,2,7,0,0,8,31,1,0,4,0,2,11,11
181	5,1,0,0,7,70,6,2,17,0,0,7,4,0,0,0,0,0,34,13
112	31,8,0,0,25,6,52,1,3,0,1,9,35,3,3,4,0,0,11,11
129	2,10,32,0,0,0,0,132,0,0,4,0,1,43,0,9,0,0,0,0
25	3,3,2,0,2,47,5,2,12,2,1,8,5,0,0,0,0,0,19,17
74	0,1,0,25,0,1,0,0,0,32,0,0,0,1,37,1,10,39,0,0
95	0,1,9,0,0,0,0,1,0,0,90,0,0,1,0,0,0,0,2,1
97	6,2,1,0,3,19,3,2,8,1,1,22,4,1,0,1,0,0,24,31
80	18,12,1,0,27,4,25,0,2,0,0,1,22,1,0,2,0,3,5,8
260	1,19,37,0,0,0,2,45,0,0,1,0,0,37,0,5,0,0,0,0
42	2,0,0,29,0,2,0,0,0,23,0,0,0,0,66,0,8,8,1,0
223	0,48,4,0,0,1,1,18,1,1,3,0,0,5,1,121,0,1,1,2
166	0,2,1,28,0,1,1,0,0,9,0,0,0,0,20,1,30,8,1,0
75	0,1,0,13,0,0,0,0,0,40,0,2,0,2,21,0,2,22,1,0
61	2,0,0,0,0,7,0,0,2,0,0,3,2,0,0,1,0,0,45,8
49	8,1,1,0,4,8,3,0,1,0,0,10,2,1,1,0,0,2,35,11

Figure 21: Results returned by calling Stored Procedure ***first_cursor_PU_time()*** for top 20 taxi zones.

In the first step, unnecessary columns such as VendorID, RateCodeID and improvement_surcharge are irrelevant, they are ignored to optimise storage and improve performance of the toolset. This is done by limiting the columns selected in Source Code 2.

```
        SELECT TaxiZoneID, TaxiZoneName FROM TaxiTrip g, TaxiZone t
```

Source Code 2: SQL select statement, selecting columns TaxiZoneID and TaxiZoneNamef rom tables TaxiTrip and TaxiZone

Abnormalities were found during the second step for taxi trip data, as Trip_distance or Total_amount being zero in some data records, which makes those records meaningless. Those data will be filtered out by the predicates in source code 3.

```
        WHERE total_amount> 0 AND trip_distance> 0
```

Source Code 3: SQL where clause statement with two predicates, filtering out records that don‘t satisfy the conditions.

The geographic information provided is incomplete as boundaries for certain taxi zones are missing, this resulted in certain taxi zones not being highlighted upon clicking.

New columns are derived by extracting hour from the pickup datatime column and group them by their pickup taxi zones in Source Code 4.

```
        WHERE HOUR (g.lpep_pickup_datetime) >= hour_1
        AND HOUR (g.lpep_pickup_datetime) < hour_2
        GROUP BY (g.PULocationID)
```

Source Code 4: SQL where clause statement with two predicates, limiting the selection of records to the conditions defined and group the results by the column selected.

For a list of complete MySQL Stored Procedures defined, please refer to Appendix 13.1.

A simple Java program was written to invoke the Stored Procedures in the database to retrieve and convert data into JSON format.

Taxi zones are polygons highly customised by TLC in .shp format, a geospatial vector data format for geographic information system. Ogre [Har16] was used to convert it into GeoJSON format.

```
  ......
   "type":"FeatureCollection",
   "features":[
      {
         "type":"Feature",
         "properties":{},
         "geometry":{
            "type":"Polygon",
            "coordinates":[
               [
                  [-74.007791, 40.74197], [.....], [.....]......
               ]
      }
  ......
```

Source Code 5: The geographic information for taxi zone Meatpacking/West Village West.

The process described above is illustrated in Figure 22. After processing the original dataset, we obtained a JSON file with all geographic information along with another JSON file that contains four data dimensions mentioned below in Section 5.2.1.

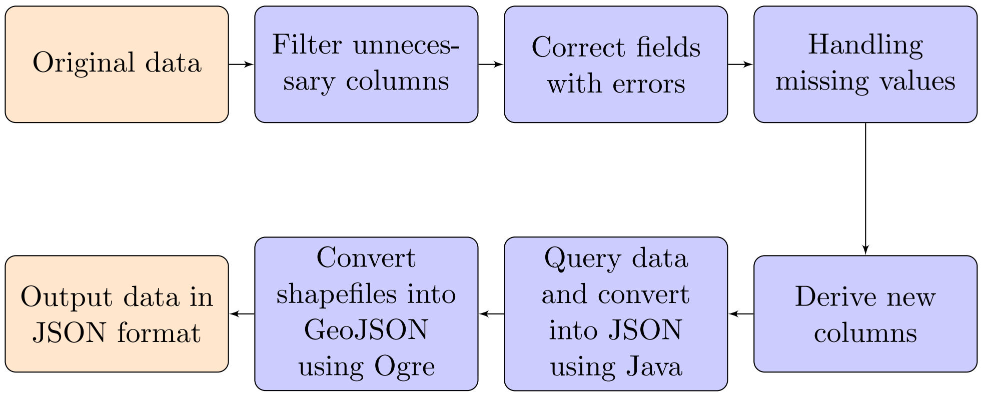

Original data
Filter unnecessary columns
Correct fields with errors
Handling missing values
Derive new columns
Query data and convert into JSON using Java
Convert shapefiles into GeoJSON using Ogre
Output data in JSON format

Figure 22: A flow chart illustrates the steps for data pre-processing.

5.2 Visualisation

This section introduces the implementation of three visualisations and emphases on the essential features with technical details.

5.2.1 Chord Diagram View

A chord diagram is ideal to visualise the taxi dataset since it is able to provide both inner-zone and inter-zone comparison.

In Smart Taxi Vis, each chord represents a taxi zone and the paths represent one data dimension. Currently there are four data dimensions available:

1. Number of taxi pickups
2. Number of taxi dropoffs
3. Average trip fares
4. Average trip distance

Data pre-processing produced an ordered list of taxi zones along with their square matrices of four data dimensions, all in JSON format. Adjusting the sliders will truncate the list and the square matrices via function ***spliceMatrix()*** and ***spliceSubMatrix()*** shown in Source Code 6.

```
function spliceMatrix(matrix) {
  matrix.splice(0, ZONE1 - 1);
  matrix.splice(ZONE2 - (ZONE1 - 1), matrix.length);
  return matrix;
}
```

```
        function spliceSubTripMatrix(matrix) {
           jQuery.each(matrix, (i, val) => {
             spliceMatrix(val);
           });
           return matrix;
        }
```

Source Code 6: Two functions used to truncate data matrices using JavaScript's array splice.

Chord Diagram View is formed by a group of Scalable Vector Graphics (SVG), the de facto standard for the construction, layout and rendering of vector images for web applications [Moz17]. Comprehensive understanding of SVG is essential before the implementation of Chord Diagram, as many interactive functions require manipulation of SVG elements. SVG are wrapped as HTML elements and thus support JavaScript function and CSS styling. SVG has a set of elements and for Chord Diagram View, elements ***svg***, ***g***, ***path***, ***text***, ***title*** and ***circle*** are used to render the visualisation and attach interactive features to it.

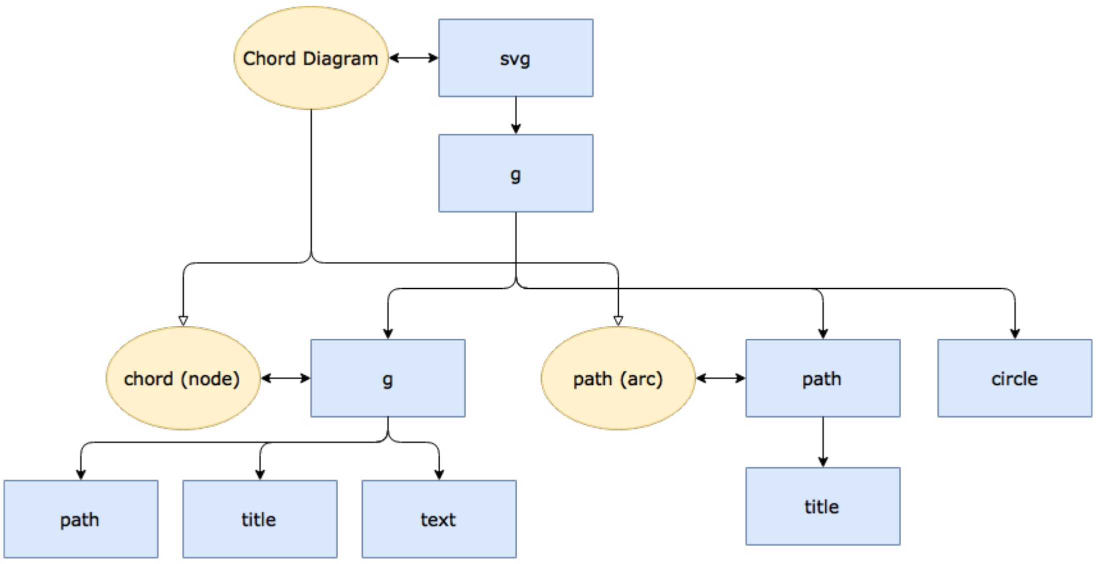


Figure 23: The hierarchy of SVG elements in Chord Diagram View.

```
......

        const outerRadius = targetSize - marginBetweenLabelAndChord;
        const innerRadius = outerRadius - chordRadiusWidth;

        const layout = d3.layout.chord();
        layout.matrix(dataMatrix);

        const arc = d3.svg.arc()
        .innerRadius(innerRadius)
        .outerRadius(outerRadius);

        const path = d3.svg.chord().radius(innerRadius);

        let chordDiagram = d3.select('#chordDiagram')
        .append('svg');
        let g = chordDiagram.append('g');

        let subG = g.append('g');
        let subGPath = subG.append('path');
        let subGPathTitle = subG.append('title');
        let subGText= subG.append('text');

        let path = g.append('path');
        let pathTitle = path.append('title');

        let circle = g.append('circle');
......
```

Source Code 7: This code snippet is compressed to show the essential logic that initialises Chord Diagram View with the hierarchy shown in Figure 23. The syntax of D3.js and jQuery can be used interchangeably is a major advantage that expedites the development.

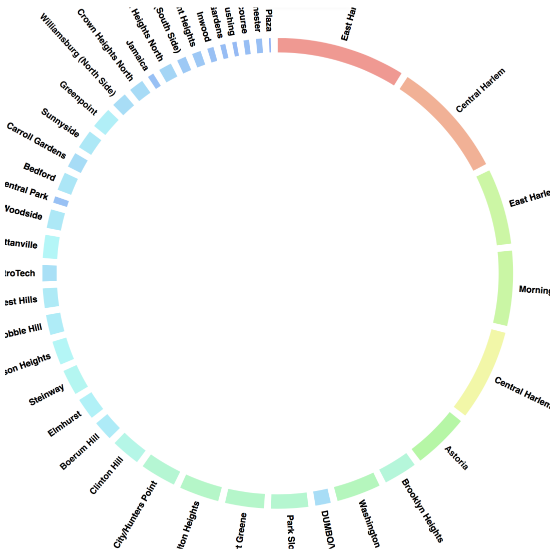


Figure 24: The chords, which are essentially a circle being sectioned into arcs.

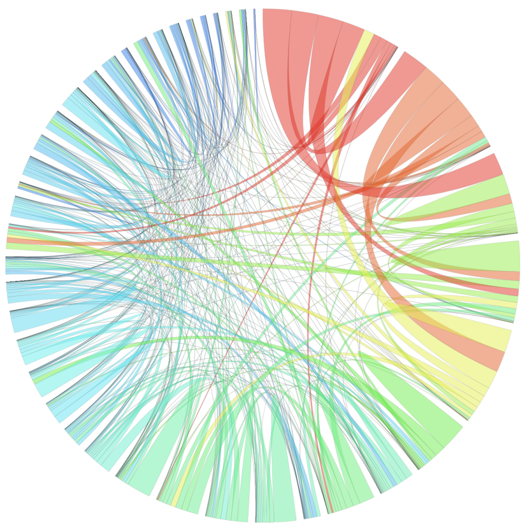


Figure 25: The paths connecting chords, the data magnitude decides their thickness.

Degrees of curvature and normalised values are calculated upon importing data matrix into ***d3.layout.chord()***. Degrees of curvature are used to render an arc and divide it into chords (see Figure 24), combining with normalised values, paths of the chords can be rendered (see Figure 25).

The colour map is dynamically generated base on the data dimension. Instead of using RGB, HSL is used. HSL stands for hue, saturation, and lightness, a HSL colour in HTML is defined as ***hsl(hue, saturation, lightness)***. Simply adjust the hue value between 0 to 220 will produce a colour on the colour map shown in Figure 26. A JavaScript function ***generateColourMap()*** in Source Code 8 that loops through the taxi zones used, inputs the corresponding count c into the formula below and returns a hue value. The loop returns a set of hue colours as hue with a set of input data c:

$$hue_i = max_{hue} - ((c_i - min_c) - (max_c - min_c)) \cdot max_{hue}$$

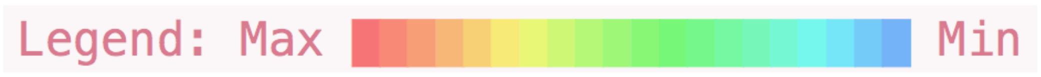
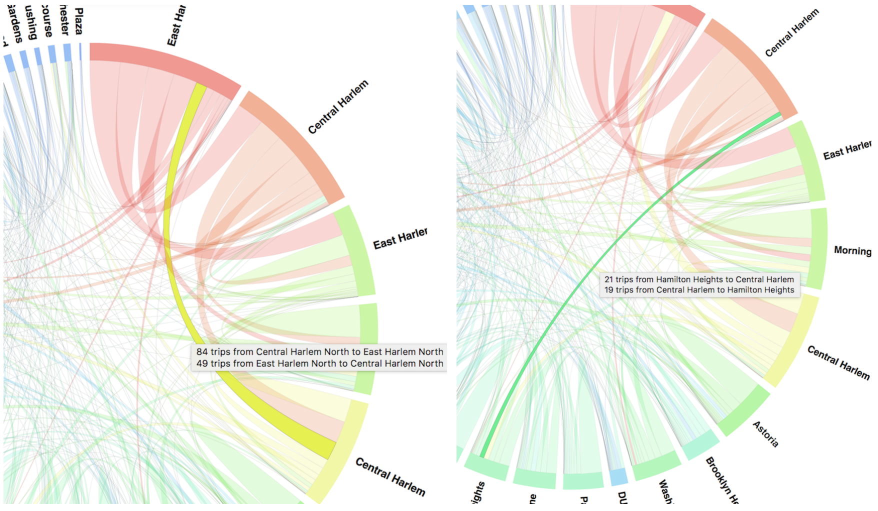


Figure 26: The colourmap legend used for Smart Taxi Vis.

The formula defines the colour of chords, it is inspired by Bumgardner's tutorial [Bum12] on making rainbow colour cycle using JavaScript. Since the data for chord diagram are always in a square matrix form, as described previously in Figure 16, the colour of paths is therefore dependant on the chord with greater data magnitude in a path connection, see illustrations in Figure 27.

```
function generateColourMap(count, maxCount, minCount) {
  const maxHue = 220;
  let hue = (count - minCount) / (maxCount - minCount);
  hue = maxHue - hue * maxHue;
  return `hsl(${hue},90%,50%)`;
}
```

Source Code 8: The function takes in count, max count and min count of the data, returns a HSL colour generated through the formula.


84 trips from Central Harlem North to East Harlem North
49 trips from East Harlem North to Central Harlem North

21 trips from Hamilton Heights to Central Harlem
19 trips from Central Harlem to Hamilton Heights

Figure 27: The colour of paths is dependant on the chord with greater data magnitude in a path connection.

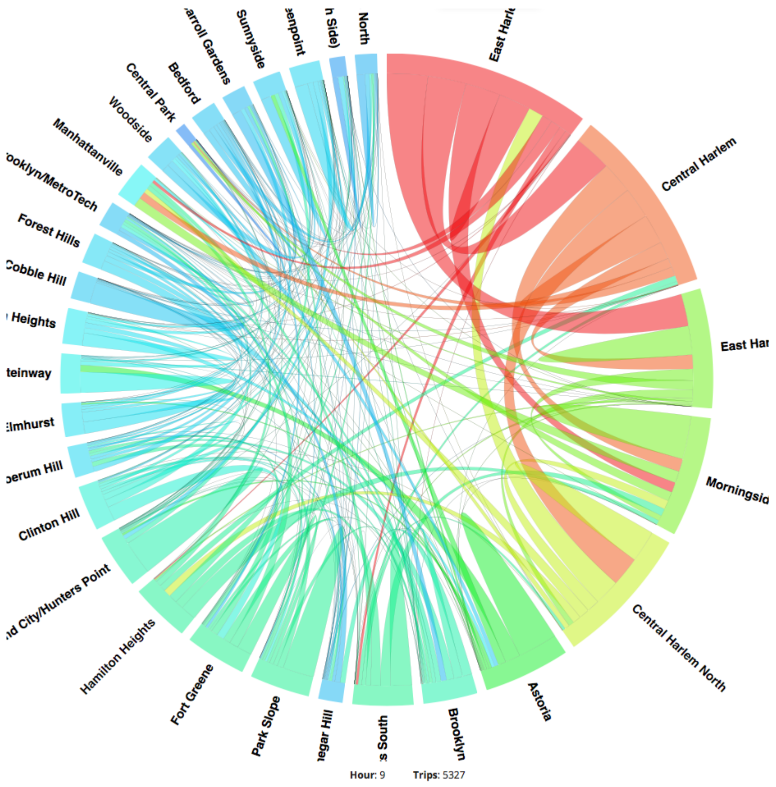


Figure 28: Chord Diagram in Smart Taxi Vis, built with D3.js [Bos15], visualising top 40 busiest taxi zones in New York, in terms of trips originated in taxi zones between 9am and 10am, for the week starting from 5th Dec 2016 to 11th Dec 2016.

5.2.2 Map View

Map View consists of four layers in total:

- Base layer (Figure 29)
- Marker layer (Figure 30)
- Connector layer (Figure 31 and 42)
- Choropleth layer (Figure 32)

After obtained the processed dataset from the previous section, a base map layer can be constructed by using the converted taxi zone polygons data in GeoJSON format. Figure 29 shows the base map obtained with Source Code 9.

```
const MAP = anychart.map();
/*Initialise an anychart map instance*/
MAP.geoData('anychart.maps.taxizone');
/*Set the geographical data directory
(the variable that stores the GeoJSON data)*/
MAP.geoIdField('LocationID');
/*Map the ID field to 'LocationID' column in GeoJSON*/
MAP.container('anymap').draw();
/*Select the HTML element with ID of anymap and draw the map*/
```

Source Code 9: The function generates and displays a base map of 263 taxi zones.

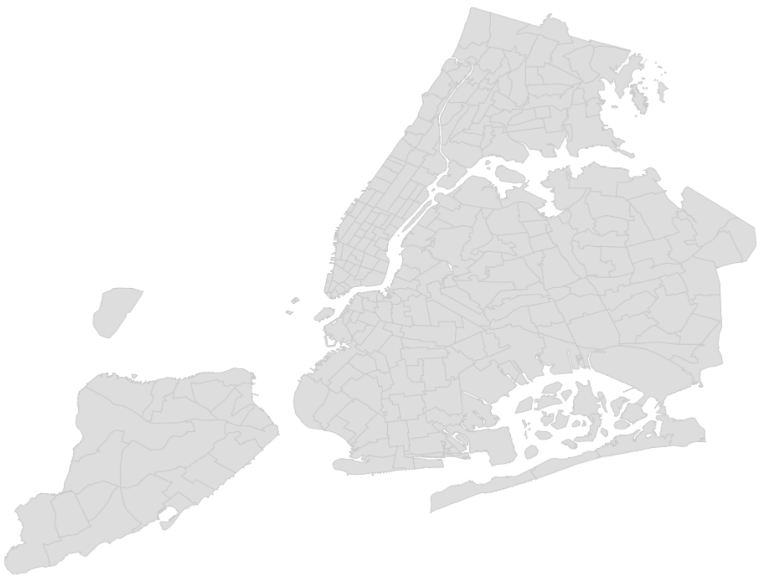


Figure 29: A base map layer of New York City built with AnyChart.js [Any17], divided into 263 taxi zones.

A marker layer as shown in Figure 30, consists of markers that represent the taxi zones being visualised in Chord Diagram, each marker is placed at the centre of the taxi zone. Since AnyChart.js does not support assigning different colours to markers within the same layer, in order to ensure colours of the marker coincide with Chord Diagram, Source Code 10 is written to overcome the limitation, instead of adding one marker layer, multiple layers are added onto the base map, therefore each taxi zone's marker will have its own colour in Map View.

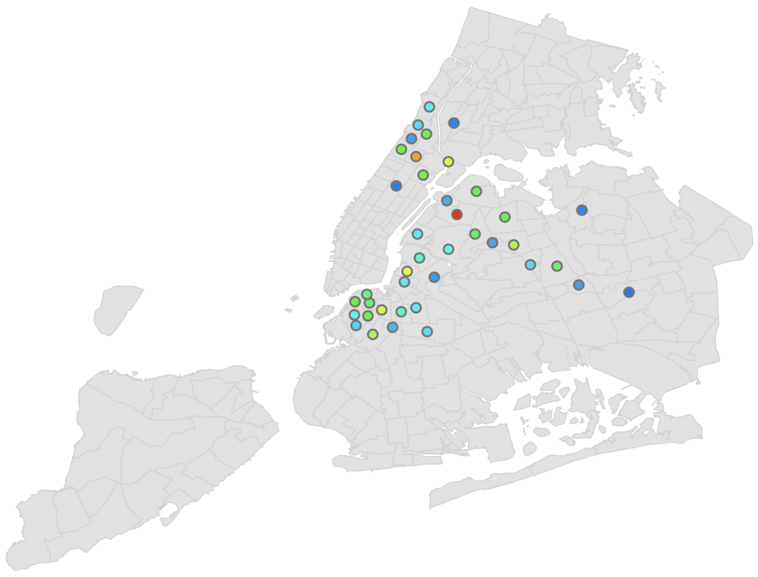


Figure 30: A marker map layer visualises top 40 busiest taxi zones in New York, in terms of trips originated in taxi zones between 9am and 10am, for the week starting from 5th Dec 2016 to 11th Dec 2016.

```
jQuery.each(ZONE_HOLDER, (i, val) => {
 const seriesData = anychart.data.set([val]);
 const newSeries = seriesData.mapAs(null, {
        name: 'ZoneName',
        id: 'ZoneId',
        size: 'Data',
        color: 'color',
 });
createMarkerSeries(val.ZoneName, newSeries, val.color);
});
```

Source Code 10: A for loop to create multiple marker layers in order to differentiate taxi zones by their colour, as shown in Figure 30.

A single-connector layer like Figure 31 contains an arrow that connects two taxi zones. In a multi-connector layer, the thickness of the arrow is mapped the corresponding data dimension while the colour is mapped to the destination's colour. Connector layer has the same limitation as marker layer, the same iteration approach used in 10 is adopted to customise connector's colour and thickness.

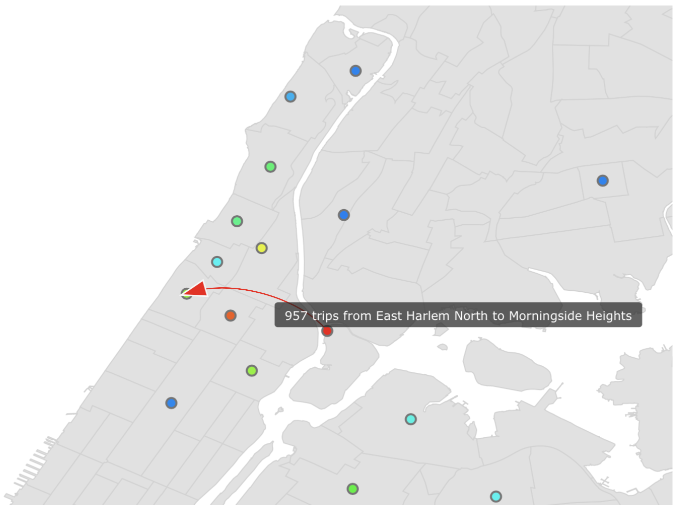


Figure 31: A connector connecting East Harlem North and Morningside Heights.

A choropleth layer contains a highlighted taxi zone on the base map. A taxi zone will be highlighted (Figure 32) upon clicking on Histogram View. A proper data pre-processing process simplifies the implementation of this function, see Source Code 11.

```
function highlightZone(zoneId) {
  removeMapSeries('highlightZone');
  const highlightZone = MAP.choropleth([{
    id: zoneId
  }]);
  highlightZone.id('highlightZone');
  highlightZone.enabled(true);
  MAP.zoomToFeature(zoneId);
}
```

Source Code 11: The function takes in the target taxi zone id and uses it to create a new choropleth layer, finally zoom in to the taxi zone.

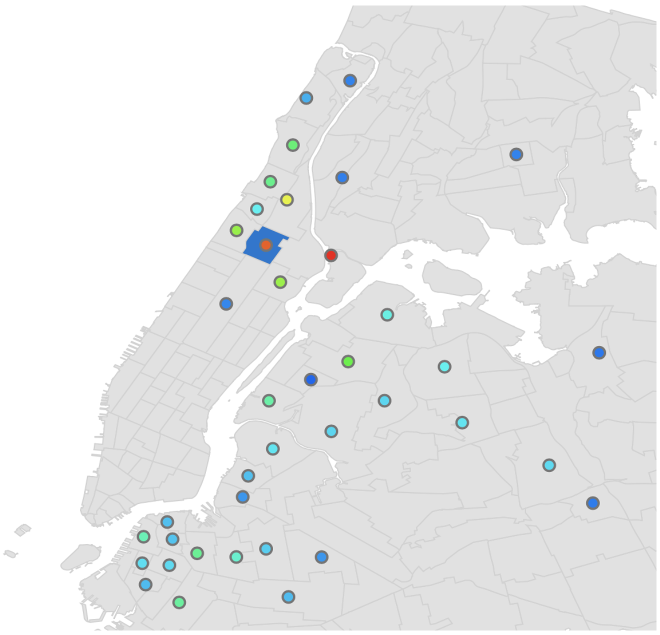


Figure 32: A taxi zone polygon is highlighted on the base map.

5.2.3 Histogram View

Histogram is a graphical representation of the distribution of numerical data. Figure 33 visualises the distribution of taxi trips in different hours of the day. Current hour is highlighted in red on the histogram.

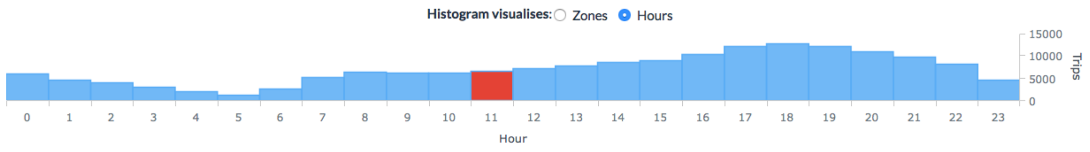


Figure 33: A histogram built with AnyChart.js [Any17], visualising the number of trips at 11am.

Histogram View also visualises the distribution of taxi trips in different taxi zones. However, since taxi zones are in a descending order, the height of bins, which is mapped to the number of trips from the taxi zone to the rest of taxi zones, are always decreasing progressively as shown in Figure 34. Therefore, a colour dimension is added to provide additional information about the data. As all three visualisations share the same colourmap legend in Figure 26, the colour of bins is used for visualising the number of taxi trips with the taxi zones currently being visualised.

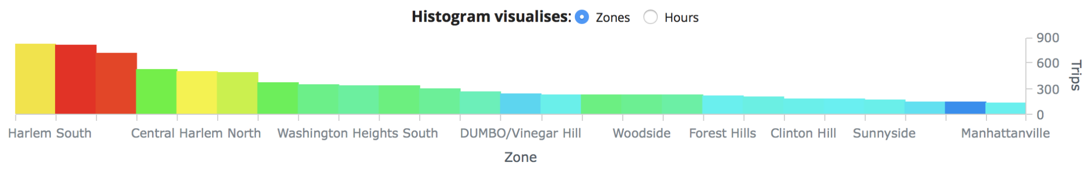


Figure 34: A histogram built with AnyChart.js [Any17], visualising the number of trips for the selected taxi zones.

5.3 Interaction and Coordination

This section introduces several ways implemented for interactions and coordinations between the visualisations presented by Smart Taxi Vis.

5.3.1 Filtering and Switching

Two sliders are used to filter the data used for generating visualisations and trigger the re-rendering of visualisations immediately. The zone slider can be slid from both end or dragged to adjust the selected range of taxi zones. The hour slider selects the specific hour of the day.

A set of radio buttons is used to switch the data dimension. Changes made to these toggles will immediately trigger the transition of all three visualisations to the selected state. Figure 37 shows the top 40 taxi zones from four different data dimensions in Chord Diagram.

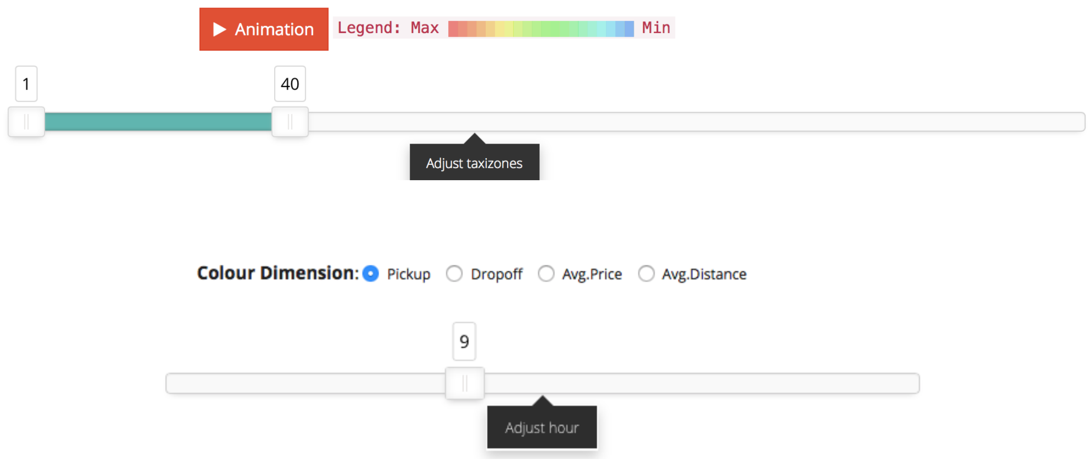
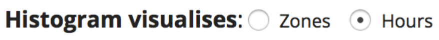


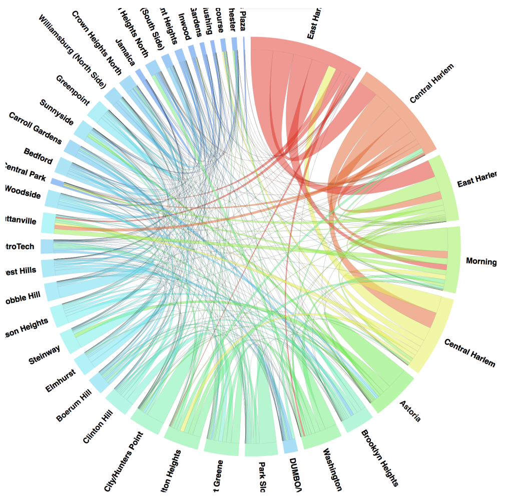
Figure 35: The area serves as a control region that consists of all toggles for all three visualisations.

Two radio buttons in Figure 36 are used for switching the data dimension of the Histogram Views shown in Figure 33 and Figure 34.

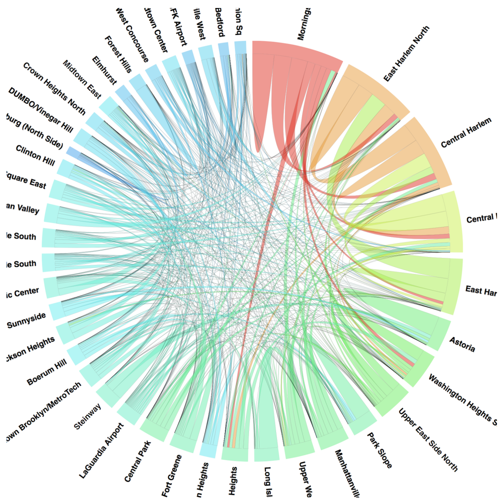


Histogram visualises: ○ Zones [✓ ● Hours]

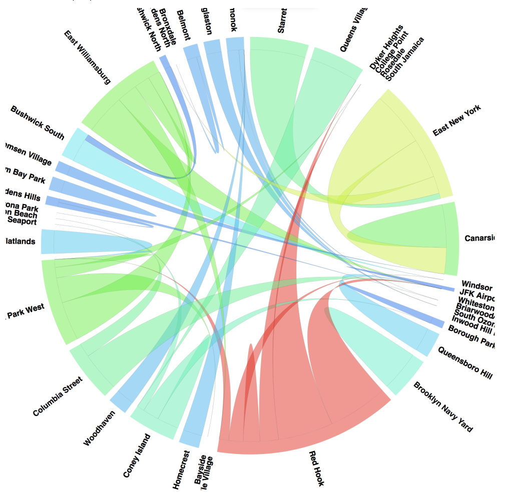
Figure 36: Radio buttons to switch the data dimension for Histogram View.



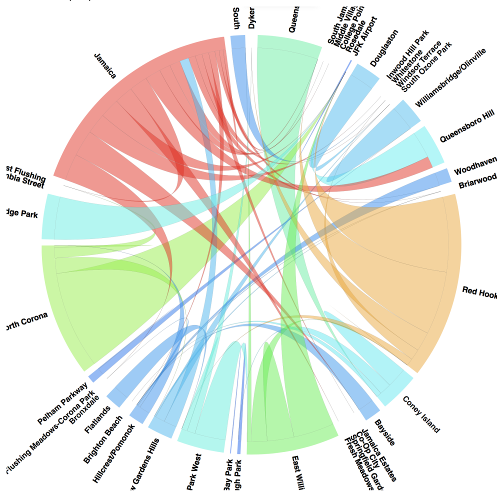
(a) Trip pickup data dimension.



(b) Trip dropoff data dimension.



(c) Average trip fare data dimension.

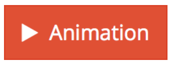


(d) Average trip distance data dimension.

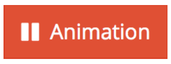
Figure 37: Example of four different data dimensions for Chord Diagram.

5.3.2 Animation and Transition

By default, there is an animation playing which loops through the data dimension by a one-hour interval. Figure 38 shows a button used to control the animation, mouse hover any visualisation will also pause the animation to allow distraction-free inspection by the user. Smooth transition is implemented for all visualisations to achieve the performance target set in the project specification.



(a) Button in pause state.



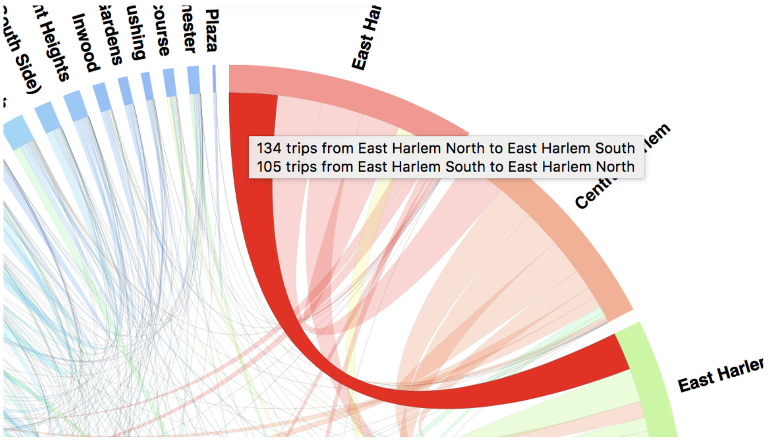
(b) Button in play state.

Figure 38: Button for toggling the animation of all visualisations.

5.3.3 Mouse Event

Three visualisations are also highly interactive with mouse hover and mouse click events, those events will then trigger coordinations between different visualisations.

In Chord Diagram, apart from displaying a tooltip with relevant information, mouse hover emphases the target path (see Figure 39) by decreasing the opacity of other paths. When hovering over a chord, only the paths belong to it will be displayed and the rest are hidden as shown in Figure 40.



134 trips from East Harlem North to East Harlem South
105 trips from East Harlem South to East Harlem North

Figure 39: Mouse hover a path in Chord Diagram.

Mouse hover in Map View triggers the displaying of a tooltip as shown in Figure 41a and Figure 41b. This is different from Chord Diagram as the tooltip shows the sum of data for all connected taxi zones, whilst Chord Diagram's chord only shows the sum of data for all visualised zones, the discrepancy is illustrated in Figure 40 and Figure41a, where the same taxi zone East Harlem North is hovered by the tooltips display two different numbers. This is due to the square matrix structure used by chord diagrams.

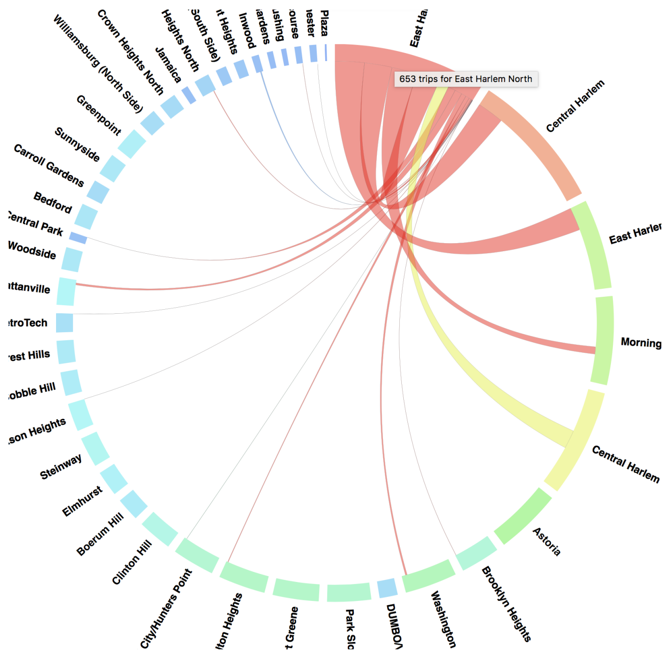
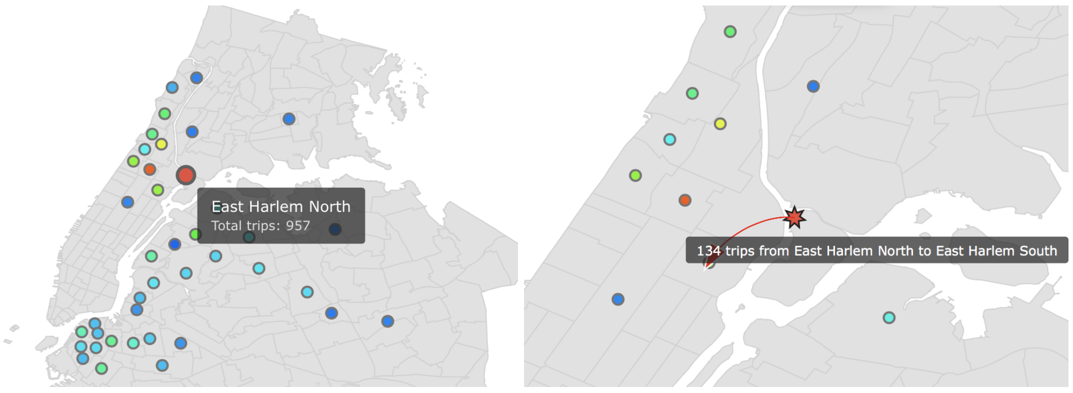


Figure 40: Mouse hover a chord in Chord Diagram.

There are coordinations between Chord Diagram and Map View. In Chord Diagram, mouse click on a path triggers the construction of a single-connector layer in Map View, connecting the origin and destination taxi zones, as shown in Figure 31. Clicking on a chord creates a multi-connector layer that connects the clicked taxi zone and destination taxi zones from it, as shown in Figure 42 below. The connectors are coordinated with the paths shown in Figure 40.



(a) Mouse hover a marker in Map View. (b) Mouse hover a conenctor in Map View.

Figure 41: Mouse hover events in Map View.

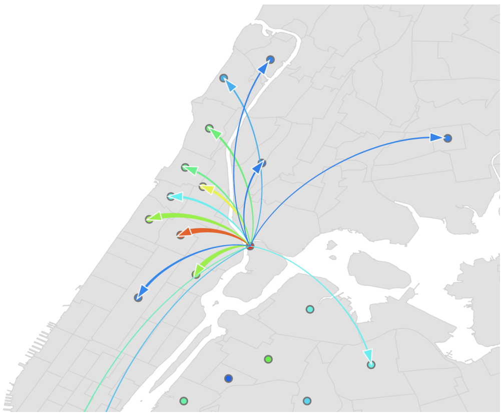


Figure 42: Connectors connecting East Harlem North to all its destinations that are currently being visualised in Map View.

Histogram View's hour view provides the same functionality as the hour slider shown in Figure 35, clicking on the bins start the transition for Chord Diagram and Map View instantly to the corresponding hour. Clicking on the bins of Histogram View's zone view zooms in on the corresponding taxi zone and highlight its entire boundary, as shown in Figure 44, while the clicked bin is highlighted in grey as shown in Figure 43.

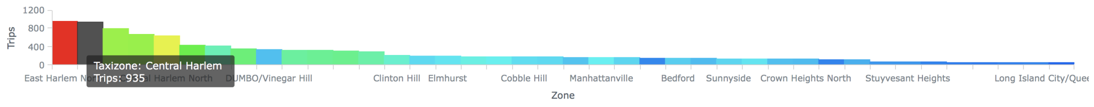


Figure 43: A bin is clicked and highlighted in grey on Histogram View's zone view.

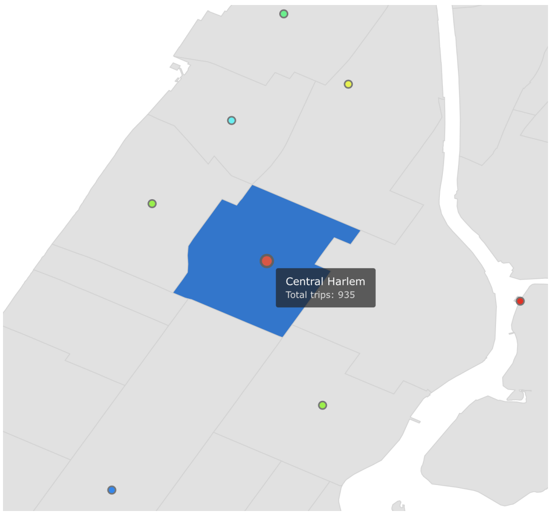


Figure 44: The taxi zone Central Harlem is highlighted on the map after the corresponding bin is clicked in Histogram View.

6 Testing

This section describes the testing of Smart Taxi Vis, with focus on functional testing, usability testing and compatibility testing.

During the development of Smart Taxi Vis, unit testing is being constantly conducted alongside the new features developed. Integration testing is performed before every git merge from development branch to master branch, to ensure the new features are integrated successfully.

6.1 Functional and Usability Testing

Three users are invited to perform functional testing and usability testing of Smart Taxi Vis. All users are asked to grade the result from both functional testing and usability testing separately, on a scale of 1 to 10 with 10 being the most positive response, results are shown in Table 5.

All users agreed that Smart Taxi Vis fulfils project specification and has all the proposed features implemented. In usability testing, user feedback are generally positive with critics from domain experts in data visualisation and website developer. Those criticisms are therefore taken into account for the future improvement for Smart Taxi Vis.

User	**Background**	**Functional Grade**	**Usability Grade**
User A	Data Visualisation	10	7
User B	Web Developer	10	7
User C	Non-related	10	8

Table 5: Users with different technical background are invited to perform functional testing and usability testing of Smart Taxi Vis, both testings are graded on a scale of 1 to 10 with 10 being the most positive response.

Feedback from user A, who has a PhD in data visualisation:

“ Chord Diagram View does an excellent job for visualising the relationship and trend of different taxi zones, in coordination with other two visualisations. However, the comparison became indistinguishable when there are too many paths (too many taxi zones being selected). Also, average distance and average price are not fully visualised by Chord Diagram View, as the chords are not representing useful information. ”

Feedback from user B, who has 5 years of web development experience with JavaScript:

“ The UI (user interface) design of Smart Taxi Vis is amateur but gets the job done. The UX (user experience) needs more work, for example, adding more tutorials or explanations of what it does and how it does. Noticeable lags can be observed when there are too many paths in Chord Diagram and too many markers in Map View, which slow down the transition as well.”

Feedback from user C, who has no relevant technical skills in both data visualisation and web development:

“ Smart Taxi Vis reveals some interesting facts about New York taxi’s operation, but there was a learning curve when it was first introduced to me. Add more user options, maybe exporting images of current visualisation could be useful for general users. Explanations in laymen’s term are needed too. ”

6.2 Compatibility Testing

Compatibility testing is mainly for desktop and laptop platform since Smart Taxi Vis requires a large screen space to exert its usefulness, mobile platform is supported to an extent with limited features.

Compatibility testings for desktop and laptop platform are performed using virtual machines created by Oracle VM VirtualBox, an open source hypervisor [Ora13].

Device	**OS**	**Browser**	**Screen Size**	**Issues**
Desktop	Windows 10	Chrome	1920x1080	None
Desktop	Windows 10	Firefox	1920x1080	None
Desktop	Windows 10	IE 11	1920x1080	None
Desktop	Ubuntu 17.04	Chrome	1920x1080	None
Desktop	macOS 10.12.6	Chrome	1920x1080	None
Laptop	Windows 10	Chrome	1366x768	None
Laptop	Windows 10	Firefox	1366x768	None
Laptop	Windows 10	IE 11	1366x768	None
Laptop	Ubuntu 17.04	Firefox	1366x768	None
Laptop	macOS 10.12.6	Safari	1366x768	None
iPad Pro	iOS 10.3.3	Safari Mobile	1366x1024	Mouse hover event not functioning.
iPhone 6	iOS 10.3.3	Safari Mobile	375x667	Mouse hover event not functioning. Significantly slowed performance.
Nexus 6P	Android 7.1.2	Chrome Mobile	412x732	Mouse hover event not functioning. Significantly slowed performance.

Table 6: List of testing environments used, covers major operating systems and modern browsers with selected desktop and laptop resolutions.

Smart Taxi Vis is compatible with all major operating systems and modern browsers, the lack of touchscreen support cripples mouse hover actions on touchscreen devices. Performance is not optimal on mobile platforms due to less amount of computing power.

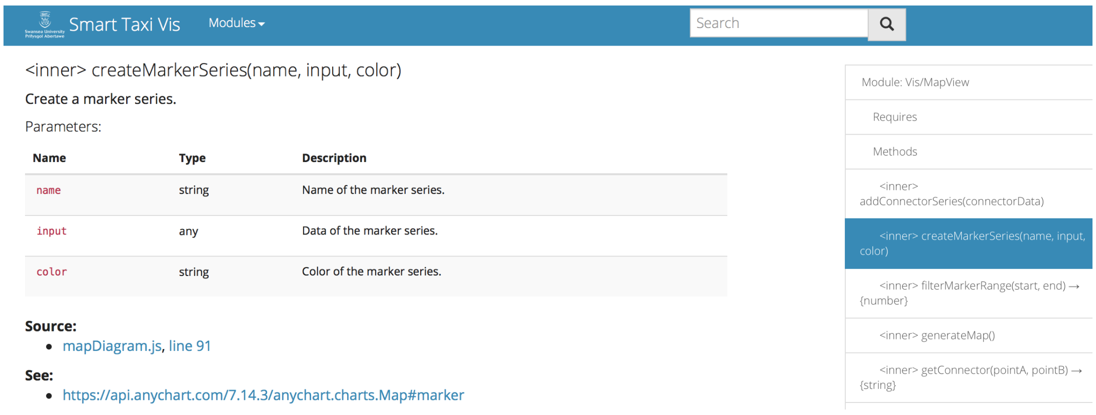
7 Documentation

The documentations is done via JSDoc, a documentation generator which produces documents base on the comment blocks in source code, as seen in Source Code 12.

For the documentation of Smart Taxi Vis, please refer to the documentation website generated by JSDoc, available along with the source code.

```
/**
 * Create a marker series.
 *
 * @param {string} name - Name of the marker series.
 * @param {any} input - Data of the marker series.
 * @param {string} color - Color of the marker series.
 * @see {@link
↪  https://api.anychart.com/7.14.3/anychart.charts.Map#marker}
 */
         function createMarkerSeries(name, input, color) {
          /* function code*/
         }
```

Source Code 12: JSDoc syntax to generate the documentation for function ***createMarkerSeries()*** shown in Figure 45.



<inner> createMarkerSeries(name, input, color)

Create a marker series.

Parameters:

Name	Type	Description
name	string	Name of the marker series.
input	any	Data of the marker series.
color	string	Color of the marker series.

Source:
- mapDiagram.js, line 91

See:
- https://api.anychart.com/7.14.3/anychart.charts.Map#marker

Module: Vis/MapView
Requires
Methods
<inner> addConnectorSeries(connectorData)
<inner> createMarkerSeries(name, input, color)
<inner> filterMarkerRange(start, end) → {number}
<inner> generateMap()
<inner> getConnector(pointA, pointB) → {string}

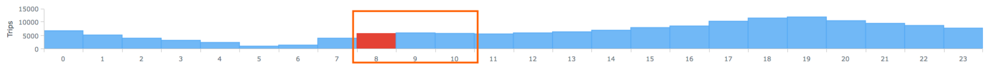
Figure 45: The documentation website contains all JavaScript functions for Smart Taxi Vis, with a search function and a navigation side panel.

For the configuration of JSDoc, please refer to Appendix 13.3.

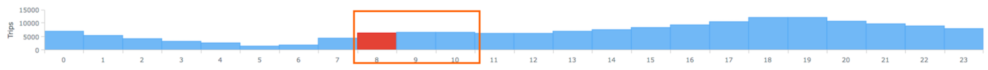
8 Observation

Through Smart Taxi Vis, the author has extracted several interesting insights from the New York Green Taxi data.

- According to the Boro Taxi Program introduced in 2013 by TLC, green taxis are not allowed to pickup passengers in Manhattan CBD areas and airport. However, pickups consistently occurred in those taxi zones albeit with a small percentage.
- No taxi surge in the morning rush hours[2] is observed, as shown in Figure 46.



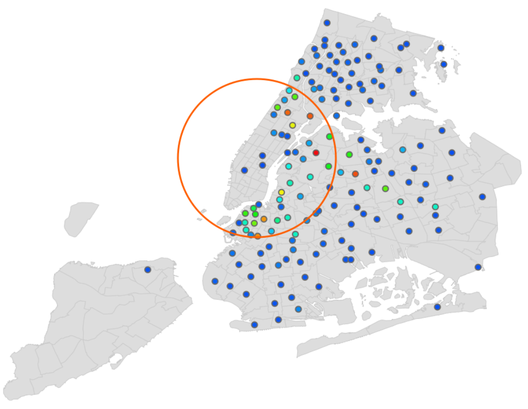
(a) Pickup data dimenstion.



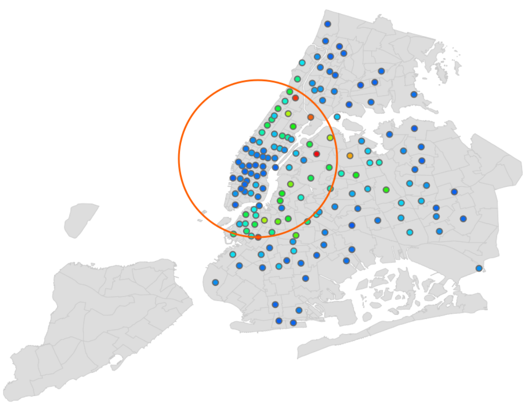
(b) Dropoff data dimenstion.

Figure 46: Histogram View for data dimension of pickup and dropoff, showing that the requirement for taxi is not surging during the morning rush hours.

- During the busiest hour, 19pm, most taxi pickups occurred around Manhattan CBD areas[3], as seen in Figure 47a. Passengers traveled generally within a close radius but slightly towards south and east, as shown in Figure 47b.



(a) Pickup data dimension with 18,281 trips occurred in 161 taxi zones, with the majority of them occurred around Manhattan CBD area, circled in the figure.



(b) Dropoff data dimension with 20,990 trips occurred in 237 taxi zones, passengers travel within a close radius of Manhattan CBD area but slightly towards south and east.

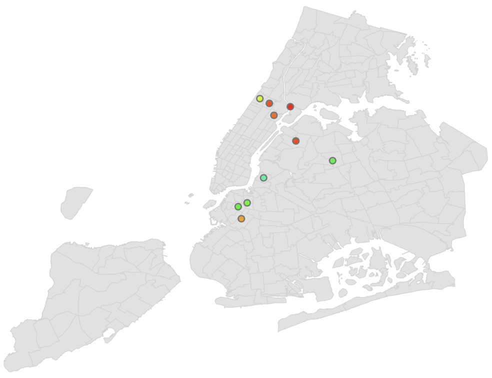
Figure 47: Map View at 19pm, the busiest hour for green taxis.

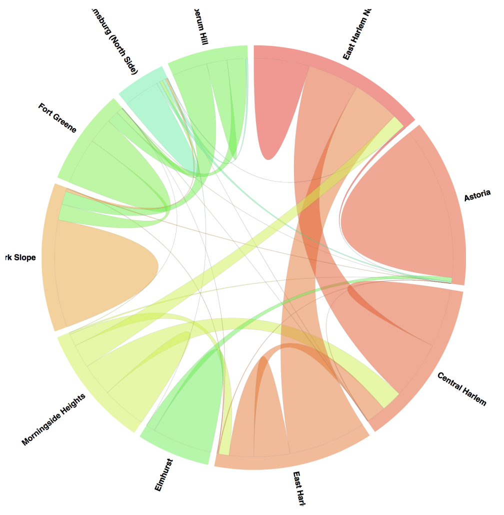
- During the busiest hour, 19pm, the top 100 taxi zones account for around 90% of taxi trips, by both pickups or dropoffs, as shown in Table 7, screenshots of the respective Map View and Chord Diagram are shown in Figure 48. Top 20 taxi zones have the highest average trips per zone for both pickups and dropoffs.

[2]From 8am to 10am
[3]Midtown Manhattan and Lower Manhattan

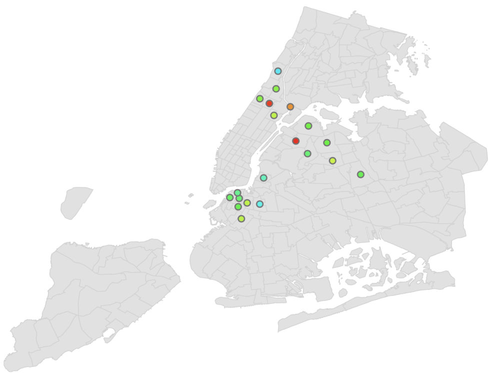
No of taxi zones	Pickups	Average	%	Dropoffs	Average	%
10	2,948	294.8	16.13%	3,283	328.3	15.64%
20	6,929	346.45	37.90%	7,105	355.25	33.85%
30	9,573	319.1	52.37%	9,853	328.43	46.94%
50	13,281	265.62	72.68%	14,120	282.4	67.27%
100	16,573	165.73	90.66%	18,399	183.99	87.65%

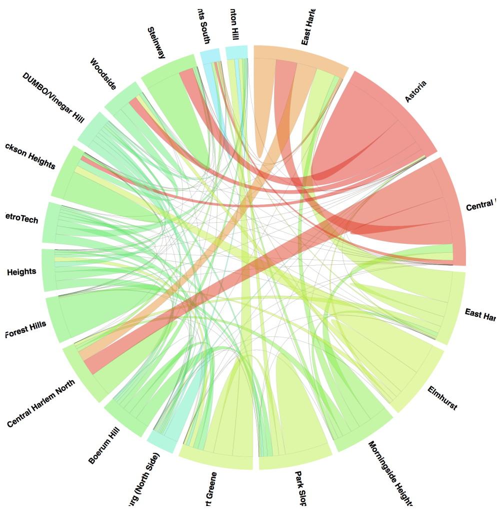
Table 7: Number of pickups/dropoffs, average trips per taxi zone and their respective percentage over the total number of trips, for different number of the busiest taxi zones selected.


(a) The top 10 taxi zones.

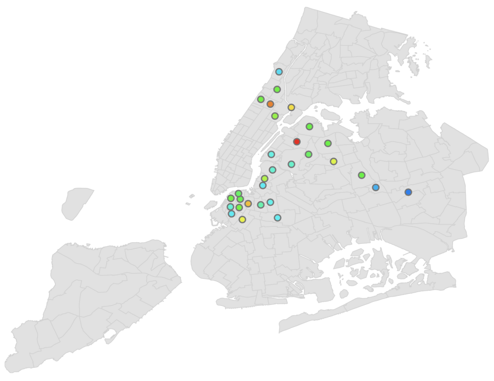


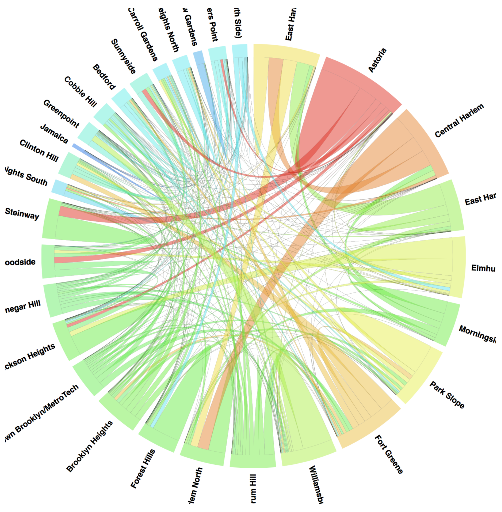
(b) The top 10 taxi zones.

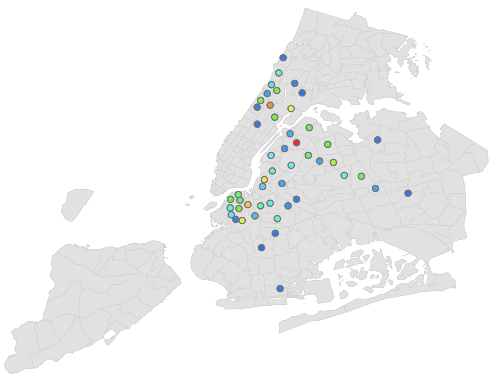

(c) The top 20 taxi zones.

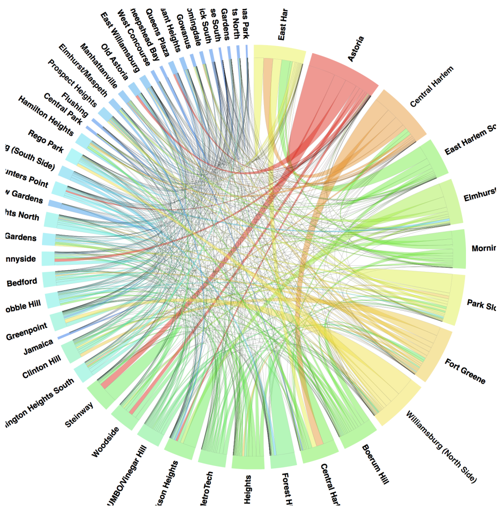


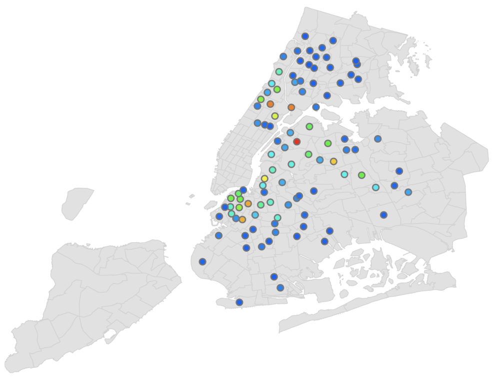
(d) The top 20 taxi zones.


(e) The top 30 taxi zones.


(f) The top 30 taxi zones.


(g) The top 50 taxi zones.


(h) The top 50 taxi zones.


(i) The top 100 taxi zones.

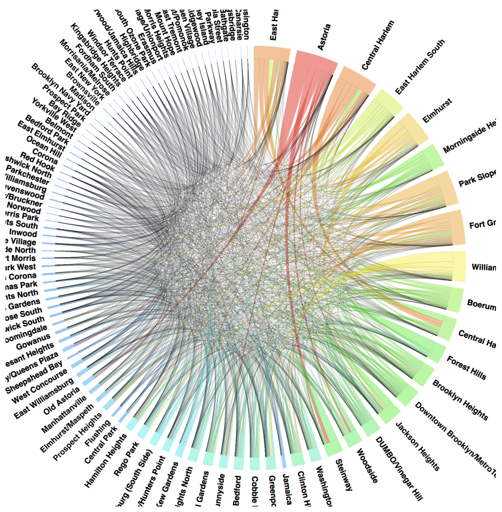
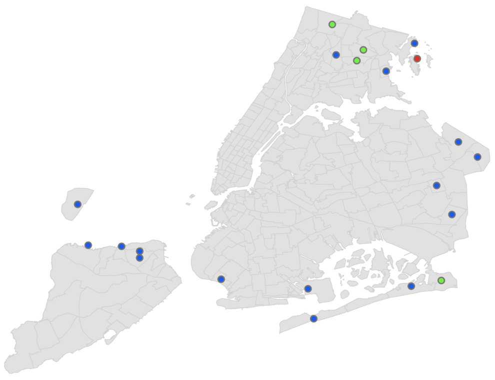
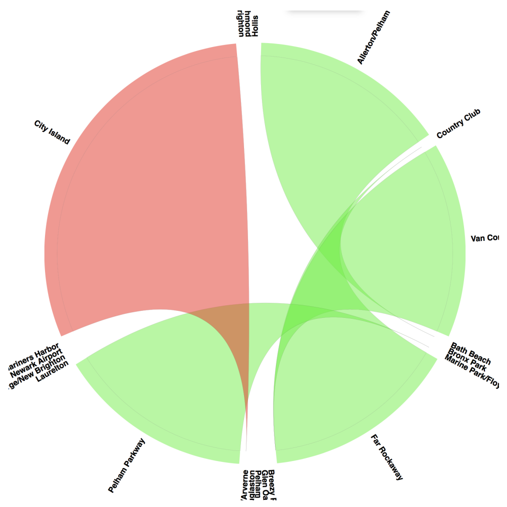

(j) The top 100 taxi zones.

Figure 48: Map View and Chord Diagram showing pickups for the respective busiest taxi zones at 19pm.

- Figure 47 shows that at 19pm, taxi trips originated from 161 taxi zones and ended in 237 taxi zones, since a chord diagram is meant to visualise the inter-relationship of taxi zones, Chord Diagram therefore will not visualise taxi zones with no trips within the zones on the diagram, Figure 49b illustrates this.



(a) The bottom 20 taxi zones.



(b) The bottom 20 taxi zones.

Figure 49: The bottom 20 taxi zones at 19pm for dropoffs, only 5 taxi zones has inter-relationship with the 20 taxi zones visualised, therefore only 5 paths in (b) are observed.

- Staten Island and Rockway Park, shown in Figure 50, have nearly no green taxi operating in the areas, despite that those regions are permitted for green taxi operation and both have residential areas.

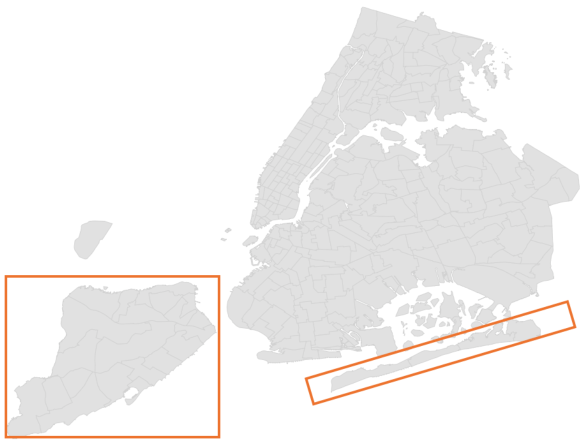


Figure 50: Staten Island is in the left orange rectangle and Rockway Park in the right one.

For more informative observations, please explore Smart Taxi Vis at `https://taxivis.wangqiru.com`. Alternatively, a demo video of major features available in Smart Taxi Vis is provided in the project folder.

9 Conclusion

Smart Taxi Vis is an interactive visualisation tool that provides visualisations generated from the taxi operational data collected by the New York City Taxi & Limousine Commission (TLC), it enables users to select and filter data dimension to present, harvesting hidden information carried by the dataset that are difficult to interpret from the huge raw dataset. A subset of the dataset was used, which is the week starting from 5th Dec 2016 to 11th Dec 2016, with 265,222 trips recorded.

Three visualisations are provided in Smart Taxi Vis:

1. Chord Diagram View that visualises the inter-relationships between entities within a given data dimension.
2. Map View, a geospatial visualisation in the form of a map with multiple layers.
3. Histogram View that visualises the distribution of data dimension.

All visualisations are highly interactive with coordinations implemented, this helps the revelation of interesting observation, such as the distribution of taxi trips over taxi zones shown in Table 7 obtained via Figure 48.

One of the key objectives of this project is for the author to gain deeper understandings of data visualisation and its application in solving real world problems, which was met through the research process, especially the literature review process and review of similar applications. The development process of this novel toolset also served as a chance to practise programming skills and data processing techniques.

10 Future Improvement

Although positive feedbacks were received from testers with different technical background, there is room for improvement and exploration for improving the usefulness of existing visualisations. The first priority is to utilise data dimensions that were left untouched in the taxi dataset, such as trip duration, number of passengers and even taxi fare payment type.

Data pro-processing currently takes a huge amount of time. This is bottlenecked by the author's knowledge in database and can be improved by optimising MySQL Stored Procedures or possibly replacing MySQL with a more suitable database.

Exploring more data dimensions denote more visualisations can be added, combined visualisations similar to those shown in Figure 8 introduced by Barry and Card [BC14] are examples to follow. More data attributes are needed to improve the usefulness of average price and average distance data dimensions, as currently they don't reveal much clear insights. A function to inspect taxi zones with a reduced level of data aggregation for geospatial data, which showa discrete location points for taxi trips, enables micro observations on the data set.

The performance of Smart Taxi Vis can be drastically tuned due to the novelty of author in JavaScript development, possibly by reconstructing JavaScript classes and introducing a framework such as Angular [Goo17]. A more scalable application can be built to accommodate the large dataset.

The user interface design and user experience design are to be overhauled, as the feedback from tester suggested. Since the author has no experience in these two area, collaboration with relevant developers with design background is a possible solution. Explanations of visualisation mapping can be added to help user better understand the information presented by Smart Taxi Vis.

11 Acknowledgements

I would like to express my gratitude to my supervisor Robert S. Laramee from Swansea University, for his guidance and support during the entire project and thesis.

My sincere appreciation is extended to Anton G. Setzer from Swansea University, for his guidance and support for project research methods.

I would like to thank the following people for the help. Liam McNabb, from Swansea University, for providing help on the topic of Smart City Visualisation, Bonie Zhang and Yang Liu, from Rutgers Business School, for providing help during the data acquisition stage and Geraldine Toh, from Nanyang Technological University, for providing help on various research matters.

I would also like to thank Swansea University for providing researching resources as well as the knowledge needed for conducting and completing this research.

12 References

[AA08] Gennady Andrienko and Natalia Andrienko. “Spatio-temporal aggregation for visual analysis of movements”. In: *VAST’08 - IEEE Symposium on Visual Analytics Science and Technology, Proceedings.* 2008, pp. 51–58. ISBN: 9781424429356. DOI: 10.1109/VAST.2008.4677356.

[Air17] Airbnb. *Airbnb JavaScript Style Guide.* 2017. URL: https://github.com/airbnb/javascript (visited on 08/15/2017).

[Any17] AnyChart. *AnyMap - Interactive JavaScript HTML5 maps.* 2017. URL: https://www.anychart.com/products/anymap/overview/ (visited on 08/16/2017).

[Arn11] Tharwon Arnuphaptrairong. “Top Ten Lists of Software Project Risks: Evidence from the Literature Survey”. In: *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS2011)* I (2011). URL: http://www.iaeng.org/publication/IMECS2011/IMECS2011_pp732-737.pdf.

[BC14] Mike Barry and Brian Card. *Visualizing MBTA Data.* 2014. URL: http://mbtaviz.github.io/ (visited on 05/16/2017).

[Boo16] Bootstrap. *Bootstrap · The world’s most popular mobile-first and responsive front-end framework.* 2016. URL: https://getbootstrap.com/docs/3.3/%20http://getbootstrap.com/ (visited on 01/01/2017).

[Bos15] Mike Bostock. *D3.JS: Data-Driven Documents.* 2015. URL: https://d3js.org/ (visited on 09/01/2017).

[Bos16] Mike Bostock. *Histogram - bl.ocks.org.* 2016. URL: https://bl.ocks.org/mbostock/3048450 (visited on 08/29/2017).

[Bum12] Jim Bumgardner. *Tutorial: Making annoying rainbows and other color cycles in Javascript.* 2012. URL: https://krazydad.com/tutorials/makecolors.php (visited on 09/11/2017).

[CDN11] Andrea Caragliu, Chiara Del Bo, and Peter Nijkamp. “Smart Cities in Europe”. In: *Journal of Urban Technology* 18.2 (2011), pp. 65–82. ISSN: 1063-0732. DOI: 10.1080/10630732.2011.601117. arXiv: arXiv:1011.1669v3.

[CGW15] Wei Chen, Fangzhou Guo, and Fei-Yue Wang. “A Survey of Traffic Data Visualization”. In: *IEEE Transactions on Intelligent Transportation Systems* 16.6 (2015), pp. 2970–2984. ISSN: 1524-9050. DOI: 10.1109/TITS.2015.2436897. URL: http://ieeexplore.ieee.org..

[Cho+11] Hafedh Chourabi et al. “Understanding Smart Cities: An Integrative Framework”. In: *Proceedings of the Annual Hawaii International Conference on System Sciences* (2011), pp. 2289–2297. ISSN: 15301605. DOI: 10.1109/HICSS.2012.615.

[Com13] NYC Taxi & Limousine Commission. *NYC Taxi and Limousine Commission - Boro Taxi program.* 2013. URL: http://www.nyc.gov/html/tlc/html/passenger/shl_passenger_background.shtml (visited on 05/19/2017).

[Cro06] Douglas Crockford. *The application/json Media Type for JavaScript Object Notation*. 2006. DOI: `10.17487/rfc4627`. (Visited on 04/26/2017).

[DA11] Mark Deakin and Husam Al Waer. "From intelligent to smart cities". In: *Intelligent Buildings International* 3.3 (2011), pp. 133–139. ISSN: 1750-8975. DOI: `10.1080/17508975.2011.586673`.

[Dox16] Doxygen. *Doxygen: Main Page*. 2016. URL: `http://www.stack.nl/~dimitri/doxygen/` (visited on 04/29/2017).

[Elt+99] L S Elting et al. "Influence of data display formats on physician investigators' decisions to stop clinical trials: prospective trial with repeated measures." In: *BMJ (Clinical research ed.)* 318.7197 (1999), pp. 1527–1531. ISSN: 0959-8138. DOI: `10.1136/bmj.319.7216.1070`.

[Esl17] Eslint.org. *ESLint - Pluggable JavaScript linter*. 2017. URL: `http://eslint.org/%20http://eslint.org/docs/about/` (visited on 08/15/2017).

[GLO12] Benedikt Gross, Joseph K. Lee, and Dietmar Offenhuber. *Boston 3-1-1 Incident Reports*. 2012. URL: `http://senseable.mit.edu/bos311/` (visited on 04/23/2017).

[Goo08] Google. *GSON*. 2008. URL: `https://github.com/google/gson` (visited on 08/29/2017).

[Goo17] Google. *Angular*. 2017. URL: `https://angular.io/` (visited on 01/01/2017).

[Har16] Marc Harter. *Ogre - ogr2ogr web client*. 2016. URL: `https://github.com/wavded/ogre` (visited on 05/01/2017).

[Joh78] S C Johnson. "Lint, a C Program Checker". In: *Comp. Sci. Tech. Rep* (1978), pp. 78–1273. ISSN: 0001-0782.

[JSD14] JSDoc. *Use JSDoc*. 2014. URL: `http://usejsdoc.org/index.html` (visited on 08/08/2017).

[Lar09] Robert S Laramee. "Bob ' s Concise Coding Conventions (C3)". In: (2009), pp. 1–11. URL: `http://cs.swansea.ac.uk/~csbob/teaching/laramee09codeConvention.pdf`.

[Lar11] Robert S. Laramee. "Bob's project guidelines: Writing a dissertation for a BSc. in computer science". In: *ITALICS Innovations in Teaching and Learning in Information and Computer Sciences* 10.1 (2011), pp. 43–54. ISSN: 14737507. DOI: `10.11120/ital.2011.10010043`.

[Liu17] Yan Liu. *Visualization of Multivariate Data*. 2017. URL: `http://people.stat.sc.edu/hansont/stat730/MultivariateDataVisualization.pdf` (visited on 04/17/2017).

[LR12] George Cristian Lazaroiu and Mariacristina Roscia. "Definition methodology for the smart cities model". In: *Energy* 47.1 (2012), pp. 326–332. ISSN: 03605442. DOI: `10.1016/j.energy.2012.09.028`.

[Map17] Mapbox. *Mapbox Studio*. 2017. URL: `https://www.mapbox.com/studio/` (visited on 04/20/2017).

[Mic15] Microsoft. *ASP.NET MVC — The ASP.NET Site*. 2015. URL: `https://www.asp.net/mvc%20http://www.asp.net/mvc` (visited on 01/01/2017).

[Mic16] Microsoft. *Visual Studio Code - Code Editing. Redefined.* 2016. URL: https://code.visualstudio.com/ (visited on 01/01/2017).

[Mic17] Microsoft. *Visual Studio IDE, Code Editor, Team Services, Mobile Center.* 2017. URL: https://www.visualstudio.com/.

[Moz17] Mozilla. *SVG — MDN.* 2017. URL: https://developer.mozilla.org/en-US/docs/Web/SVG (visited on 06/05/2017).

[NYC] NYC Taxi & Limousine Commission. *NYC Taxi & Limousine Commission - Taxicab Passenger Enhancements Project (TPEP) Archive.* URL: http://www.nyc.gov/html/tlc/html/industry/taxicab_serv_enh_archive.shtml (visited on 04/24/2017).

[NYC15] NYC Taxi & Limousine Commission. *Data Dictionary - SHL Trip Records.* 2015. URL: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml. (visited on 04/25/2017).

[NYC17] NYC Taxi & Limousine Commission. *NYC Taxi & Limousine Commission - Trip Record Data.* 2017. URL: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml (visited on 04/24/2017).

[Ora13] Oracle Corporation. "Oracle VM VirtualBox". In: *User Manual* (2013), pp. 1–346. URL: https://www.virtualbox.org/.

[Ora15] Oracle Corporation. *MySQL Workbench 6.3.* 2015. URL: https://www.mysql.com/products/workbench/%20mysqlworkbench.org.

[Piv17] Pivit Inc. *Zube — Agile project management with a seamless GitHub integration.* 2017. URL: https://zube.io/ (visited on 04/30/2017).

[Pu+13] Jiansu Pu et al. "T-watcher: A new visual analytic system for effective traffic surveillance". In: *Proceedings - IEEE International Conference on Mobile Data Management.* Vol. 1. 2013, pp. 127–136. ISBN: 978-0-7695-4973-6. DOI: 10.1109/MDM.2013.23.

[Rat+16a] Carlo Ratti et al. *MONITOUR.* 2016. URL: http://senseable.mit.edu/monitour-app/ (visited on 04/23/2017).

[Rat+16b] Carlo Ratti et al. *MONITOUR.* 2016. URL: http://senseable.mit.edu/monitour-app/%7B%5C#%7Dwa551396 (visited on 04/23/2017).

[Rib13] Severino Ribecca. *Chord Diagram.* 2013. URL: http://www.datavizcatalogue.com/methods/chord_diagram.html (visited on 08/29/2017).

[Sof17] Software Freedom Conservancy. *Git.* 2017. URL: https://git-scm.com/ (visited on 04/30/2017).

[Uni14] Population Department United Nations, Department of Economic and Social Affairs. *World Urbanization Prospects.* Tech. rep. 9. 2014, p. 32. DOI: 10.4054/DemRes.2005.12.9. arXiv: arXiv:1011.1669v3. URL: https://esa.un.org/unpd/wup/publications/files/wup2014-highlights.Pdf.

[Vli07] Hans Van Vliet. *Software Engineering: Principles and Practice.* Wiley, 2007.

[Wan17] Qiru Wang. “Smart City Visualisation : Interactive Data Visualisation for New York Taxi Operational Data”. In: *CSCM10 - Computer Science Project Research Methods. Swansea University Department of Computer Science.* (2017).

[WGK10] MO Ward, G Grinstein, and D Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications.* 2010, pp. 1–5. ISBN: 9781482257380.

[Yue+09] Yang Yue et al. “Mining time-dependent attractive areas and movement patterns from taxi trajectory data”. In: *2009 17th International Conference on Geoinformatics, Geoinformatics 2009.* IEEE, Aug. 2009, pp. 1–6. ISBN: 9781424445639. DOI: `10.1109/GEOINFORMATICS.2009.5293469`. URL: `http://ieeexplore.ieee.org/document/5293469/`.

[Zen+13] Wei Zeng et al. “Visualizing interchange patterns in massive movement data”. In: *Computer Graphics Forum* 32.3 PART3 (June 2013), pp. 271–280. ISSN: 14678659. DOI: `10.1111/cgf.12114`. URL: `http://doi.wiley.com/10.1111/cgf.12114`.

13 Appendix

13.1 MySQL Stored Procedures

The following Store Procedures orders taxi zones by pickups and grouped them hourly.

```
CREATE PROCEDURE `first_cursor_PU_time` (IN user_limit INT,IN time_1
↪  INT,IN time_2 INT) DETERMINISTIC BEGIN DECLARE vDone INT DEFAULT 0;
↪  DECLARE vId INT; DECLARE vPU INT; DECLARE vDO INT; DECLARE cursor1
↪  CURSOR FOR
SELECT T1.TaxiZoneID FROM (
SELECT TaxiZoneID,COUNT(*) AS PUNumbers FROM TaxiTrip g,TaxiZone t
↪  WHERE g.PULocationID=t.TaxiZoneID AND HOUR
↪  (g.lpep_pickup_datetime)>=time_1 AND HOUR (g.lpep_pickup_datetime)<
↪  time_2 AND total_amount> 0 AND trip_distance> 0 GROUP BY
↪  g.PULocationID) T1 INNER JOIN (
SELECT TaxiZoneID,COUNT(DOLocationID) AS DONumbers FROM TaxiTrip
↪  g,TaxiZone t WHERE g.DOLocationID=t.TaxiZoneID AND HOUR
↪  (g.lpep_dropoff_datetime)>=time_1 AND HOUR
↪  (g.lpep_dropoff_datetime)< time_2 AND total_amount> 0 AND
↪  trip_distance> 0 GROUP BY g.DOLocationID) T2 ON
↪  T1.TaxiZoneID=T2.TaxiZoneID ORDER BY T1.PUNumbers DESC LIMIT
↪  user_limit; DECLARE CONTINUE
HANDLER FOR NOT FOUND
SET vDone=1;
DROP TEMPORARY TABLE IF EXISTS tblResults;
CREATE TEMPORARY TABLE IF NOT EXISTS tblResults (`TaxiZoneID`
↪  INT,`matrix` VARCHAR (5000)); OPEN cursor1; curloop : LOOP FETCH
↪  cursor1 INTO vId; IF vDone THEN LEAVE curloop; END IF;
CALL second_cursor_PU_time (vId,user_limit,time_1,time_2,@re);
INSERT INTO tblResults VALUES (vId,@re); END LOOP; CLOSE cursor1;
SELECT*FROM tblResults;
END
```

```
CREATE PROCEDURE `second_cursor_PU_time` (IN PU INT,IN user_limit
↪  INT,IN time_1 INT,IN time_2 INT,OUT json VARCHAR (5000)) BEGIN
↪  DECLARE vDone INT DEFAULT 0; DECLARE DOID INT; DECLARE count_v INT;
↪  DECLARE counter INT; DECLARE cursor1 CURSOR FOR
SELECT PULocationID,count(PULocationID) AS c FROM TaxiTrip WHERE HOUR
↪  (lpep_pickup_datetime)>=time_1 AND HOUR (lpep_pickup_datetime)<
↪  time_2 AND total_amount> 0 AND trip_distance> 0 GROUP BY
↪  PULocationID ORDER BY c DESC LIMIT user_limit;
DECLARE CONTINUE
HANDLER FOR NOT FOUND
SET vDone=1;
SET json=""; OPEN cursor1; curloop : LOOP FETCH cursor1 INTO
↪  DOID,count_v; IF vDone THEN LEAVE curloop; END IF;
SELECT COUNT(PULocationID) INTO @counter FROM TaxiTrip WHERE
↪  PULocationID=PU AND DOLocationID=DOID AND HOUR
↪  (lpep_pickup_datetime)>=time_1 AND HOUR (lpep_pickup_datetime)<
↪  time_2 AND total_amount> 0 AND trip_distance> 0;
SET json=CONCAT(json,cast(@counter AS CHAR (100)),",");
END LOOP; CLOSE cursor1;
SET json=CONCAT(LEFT (json,length(json)-1),"");
END
```

```
CREATE PROCEDURE `getZoneByPU` (IN user_limit INT,IN time_1 INT,IN
↪  time_2 INT) DETERMINISTIC BEGIN DECLARE vDone INT DEFAULT 0;
↪  DECLARE vTaxiZoneID INT; DECLARE vTaxiZoneName VARCHAR (50);
↪  DECLARE vPUNumbers INT; DECLARE cursor1 CURSOR FOR
SELECT TaxiZoneID,TaxiZoneName,COUNT(*) AS PUNumbers FROM TaxiTrip
↪  g,TaxiZone t WHERE g.PULocationID=t.TaxiZoneID AND HOUR
↪  (g.lpep_pickup_datetime)>=time_1 AND HOUR (g.lpep_pickup_datetime)<
↪  time_2 AND total_amount> 0 AND trip_distance> 0 GROUP BY
↪  g.PULocationID ORDER BY PUNumbers DESC LIMIT user_limit;
DECLARE CONTINUE
HANDLER FOR NOT FOUND
SET vDone=1;
DROP TEMPORARY TABLE IF EXISTS tblResults;
CREATE TEMPORARY TABLE IF NOT EXISTS tblResults (`TaxiZoneID`
↪  INT,`TaxiZoneName` VARCHAR (50),`PUNumbers` INT); OPEN cursor1;
↪  curloop : LOOP FETCH cursor1 INTO
↪  vTaxiZoneID,vTaxiZoneName,vPUNumbers; IF vDone THEN LEAVE curloop;
END IF;
INSERT INTO tblResults VALUES (vTaxiZoneID,vTaxiZoneName,vPUNumbers);
END LOOP; CLOSE cursor1;
SELECT*FROM tblResults;
END
```

For the rest of Store Procedures, please refer to TaxiData.sql in /Development/MySQL_StoreProcedures folder.

13.2 Coding Frequency (Git) Visualisations

GitHub provides some interesting information about the development process of Smart Taxi Vis via data visualisation.

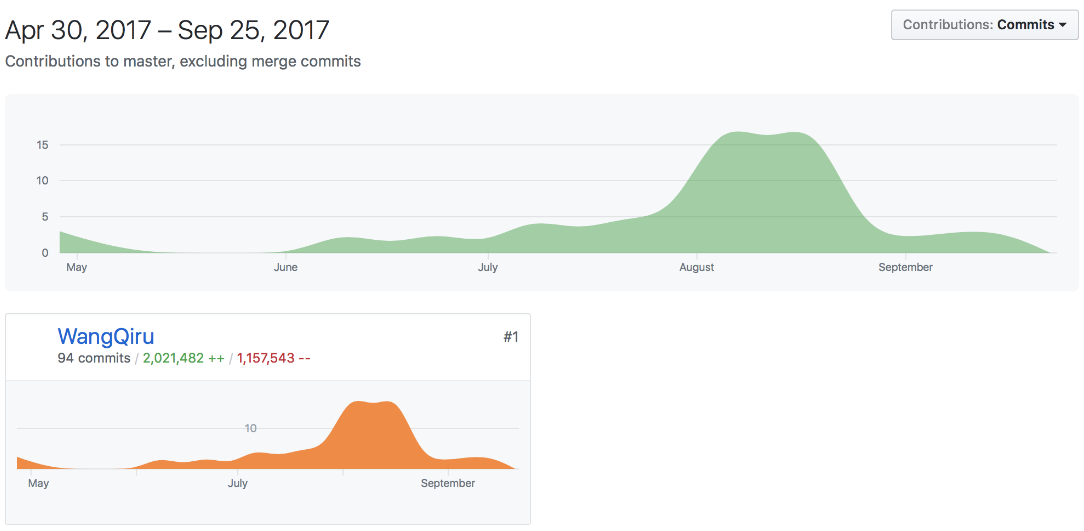

Apr 30, 2017 – Sep 25, 2017
Contributions to master, excluding merge commits
Contributions: Commits
15
10
5
0
May
June
July
August
September
WangQiru
#1
94 commits / 2,021,482 ++ / 1,157,543 --
10
May
July
September

Figure 51: GitHub commit frequency, from 30th April to 25th Sep.

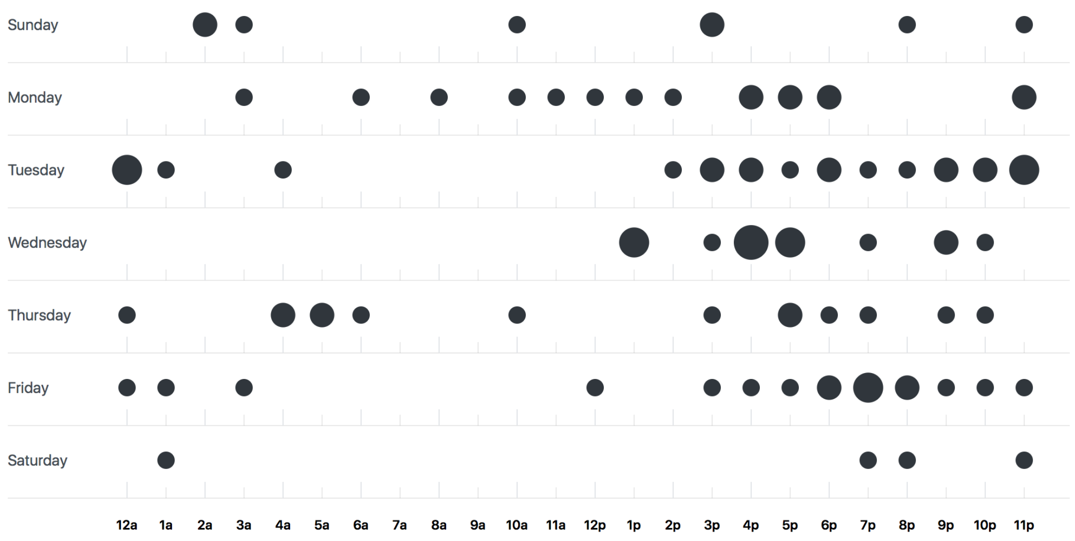


Figure 52: GitHub commits by day of week, from 30th April to 25th Sep.

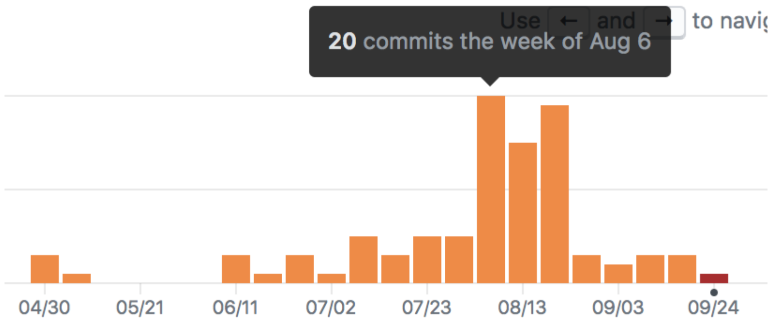


Figure 53: GitHub commits by week, from 30th April to 25th Sep.

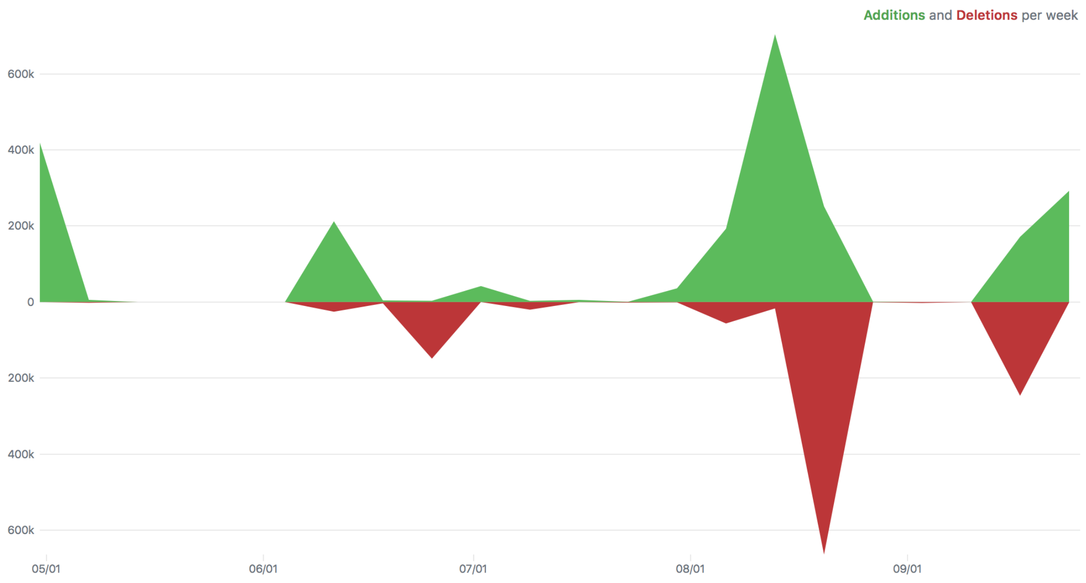


Figure 54: GitHub code additions and deletions, from 30th April to 25th Sep.

13.3 JSDoc Configuration

Below in Source Code 13 is the configuration of JSDoc used for generating the documentation of Smart Taxi Vis.

```
{
  "tags": {
    "allowUnknownTags": true
  },
  "plugins": ["plugins/markdown"],
  "templates": {
    "logoFile": "img/swansea-university-logo-white.png",
    "cleverLinks": false,
    "monospaceLinks": false,
    "dateFormat": "ddd MMM Do YYYY",
    "outputSourceFiles": true,
    "outputSourcePath": true,
    "systemName": "Smart Taxi Vis",
    "footer": "",
    "copyright": " 2017 - Smart Taxi Vis - Qiru Wang",
    "navType": "vertical",
    "theme": "yeti",
    "linenums": true,
    "collapseSymbols": false,
    "inverseNav": true,
    "protocol": "html://",
    "methodHeadingReturns": true
  },
  "recurseDepth": 10,
  "source": {
    "include": ["../js/"],
    "exclude": ["../js/lib/", "../js/data/"],
    "includePattern": ".+\\.js$"
  },
  "markdown": {
    "parser": "gfm",
    "hardwrap": true
  }
}
```

Source Code 13: JSDoc configuration to generate the documentation for Smart Taxi Vis.

13.4 Minutes of Meetings

Weekly meeting was held during this research with supervisor Robert S. Laramee. Below is one of the meeting minutes recorded.

```
Minutes of Meeting: Bob, Wang, Elif, Xiaoxiao
----------------------------------------------
Date:        25 August 2017
Start time: 15:00
End time:    17:10

Date and time of next meeting: 31 August 2017 13:00

Topics discussed:
    -- Intellij by JetBrains
    -- Bob is away 1st week of September
```

```
Progress:
    --  Elif: Watched first five lectures until end
    --  Elif: Re-structered code
    --  Xiaoxiao: Class version
    --  Xiaoxiao: Next version
    --  Wang: Read taxi cost distance read abd stored
    --  Wang: Read next version of color map

TODO:
    --  For future reference pseudo code in minutes should like pseudo
     ↪  code when type up
    --  Elif+Xiaoxiao: Keep watching software engineering lectures
    --  Elif: Introduce a class called tuple that contains information
     ↪  about 1 week- ArrayList<tuple>
    --  Elif: State object contains information about 1
     ↪  state-Arraylist<state>
    --  Elif: Introduce a region object that contains information about
     ↪  1 week- ArrayList<region>
    --  Elif: Render regions+state in treemap-see pseudo code
    --  Xiaoxiao: No magic numbers
    --  Xiaoxiao: Change TopWords to Concordance
    --  Xiaoxiao: Next version of
    --  Wang: Try adding blue to color legends
    --  Wang: Click on chord and highlight zones on map
    --  Wang: Find out what's going on with Corona zone
    --  Wang: Update on-mouse-over labels to reflect current user
     ↪  option selection
    --  Wang: User option-user clicks on zones and sees arrow to all
     ↪  obstinations
    --  Elif + Xiaoxiao: Continue watching software engineering
     ↪  lectures


    --  Elif (pseudocode):
      Treemapper: ReadData()
         while reading each line
         Tuple  tuple= new Tuple();
         Set Tuple Value
         Sort Tuple By State Name
         Compute totals Per State()
         Compute Totals per Region()
     //End Read Data()

       Render()
    ---------------------
         Draw Region()  // easy
         Draw State()
         DrawRegion()
           For each region
            DrawRegion
           End for
        EndDrawRegion()

         DrawRegion()
           Draw Outer Rectangle
           For each state in region
              DrawState()
           End for
         End Draw Region()

    --  Xiaoxiao (pseudocode):

        TranslationVisualization //visualization--name of program

        List<version> m_Versions //an array of version objects

        main method{
                newProgram();
                readData();
                newFame();
                newPanel();
                new versions();
        }

        Object:
        Version  // stores information about one version/translation
                <String> m_words  //new array of words
                m_concordance aconcordance object

        Object:
        Concordance  //store information about one concordance
                <String> m_Topwords //new array of topwords

        Procedure:
        ReadData(){
                for each file
                        v= new version(); //16 versions
                           //store new words to version
                            //compute concordance
         }
```

In total, 22 meetings were conducted during the entire project, for the rest of meeting minutes, please visit `http://cs.swansea.ac.uk/~cswang/minutes/`.