Journal of Visual Languages and Computing 41 (2017) 41–53

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

A viewpoint based approach to the visual exploration of trajectory

Jie Li[a,c,*], Zhao Xiao[a], Jun Kong[b]

[a] School of Computer Software, Tianjin University, Tianjin, PR China
[b] Department of Computer Science, North Dakota State University, Fargo, 58105, USA
[c] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210093, PR China

ARTICLE INFO

Article history:
Received 2 September 2015
Revised 9 May 2016
Accepted 5 April 2017
Available online 10 April 2017

Keywords:
Viewpoint
Spatiotemporal visualization
Trajectory
Google earth
KML
Visual analytics

ABSTRACT

We present a new viewpoint-based approach to improving the exploration effects and efficiency of trajectory datasets. Our approach integrates novel trajectory visualization techniques with algorithms for selecting optimal viewpoints to explore the generated visualization. Both the visualization and the viewpoints will be represented in the form of KML, which can be directly rendered in most of off-the-shelf GIS platforms. By playing the viewpoint sequence and directly utilizing the components of GIS platforms to explore the visualization, the overview status, detailed information, and the time variation characteristics of the trajectories can be quickly captured. A case study and a usability experiment have been conducted on an actual public transportation dataset, justifying the effectiveness of our approach. Comparing with the basic exploration approach without viewpoints, we find our approach increases the speed of information retrieval when analyzing trajectory datasets, and enhances user experiences in 3D trajectory exploration.

1. Introduction

With the increasing use of GPS devices, huge amounts of trajectory datasets are generated from different types of moving objects equipped with GPS sensors. Due to the spatial and temporal aspects of the moving objects and quantitative attributes about the encompassing environment of the routes, trajectories have become valuable sources of a number of applications, such as planning public transportation and analyzing human behavior.

Among all of the analysis requirements, directly exploring all the three aspects (space, time, attribute) of multiple trajectories is the most basic and important one, through which analysts can obtain high level characteristics of the entire datasets and preliminarily select a subset satisfying a specified filtering condition from huge amounts of raw data for further in-depth studies. An intuitive and effective exploration of trajectory is the foundation of performing different types of analysis tasks.

Trajectory simultaneously has spatial, temporal and multiple attribute dimensions, and the dependencies between them should be jointly analyzed to gain insight into the spatiotemporal dynamics of attributes. Most existing trajectory exploration/visualization techniques utilize aggregation or clustering to reduce the complexity (both amounts and dimensions) of raw datasets to arrange the visualization in a 2D plane. Information loss caused by aggregation and clustering makes it difficult to explore all the aspects of multiple trajectories in a plane view.

Therefore, 3D trajectory visualization techniques have become a common strategy of trajectory visualization. In most cases of 3D trajectory visualization, visual designers often utilize the vertical dimension to represent the temporal axis and stack multiple trajectories having the same routes [1–3]. Although the effectiveness of 3D visualization techniques on jointly displaying spatial, temporal, and attributes of trajectory data has been confirmed [4,5], they have the following two problems, which have not been well resolved in previous works: (1) Projecting 3D visualization objects in a 2D screen results in overlapping and clutter. The objects in the front hinders the user from observing other objects behind (seeFig. 1a and b). (2) Exploration parameters in 3D visualization space, such as the viewing direction, viewing distance, and external lighting, affect the display effect and the cognition speed. In an actual scene, to explore a customized 3D view containing lots of visualization objects (see Fig. 1c), the user has to manually change the exploration parameters by frequently utilizing a mouse to drag the screen to obtain an optimized display effect. Such two problems make the exploration in 3D space more time-consuming and error-prone than that in 2D space, because screen is frequently switched among different parts of the visualization to determine the findings. This problem is extremely serious for large datasets

* Corresponding author. Tianjin University, 135 Yaguan Road, Jinnan District, Tianjin, 300350, PR China.
E-mail addresses: vassilee@tju.edu.cn, jie.li@tju.edu.cn (J. Li).

http://dx.doi.org/10.1016/j.jvlc.2017.04.001
1045-926X/

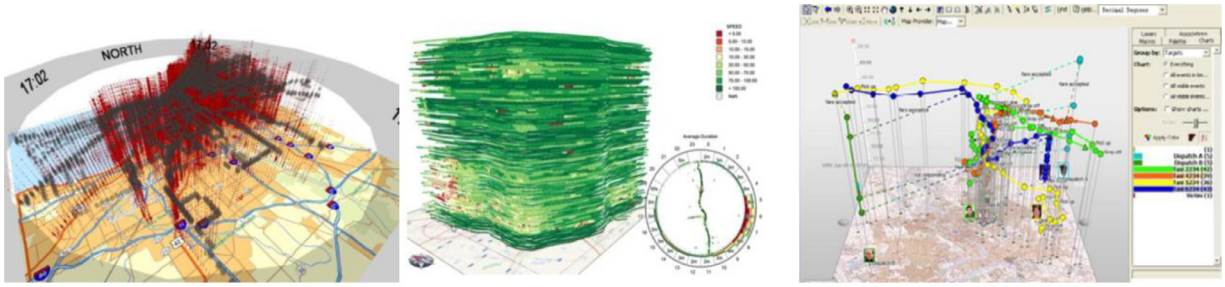


Fig. 1. Three 3D trajectory visualization techniques: (a) Space-Time Cloud [6] with a clutter problem when showing too much data. (b) Trajectory wall [1] preventing the users from viewing the objects behind the wall. (c) GeoTime [7] in which the viewpoint sequence and the angle of the sight line affect the display effects.

due to the fact that it is usually unclear where interesting patterns can be found and which trajectories needs to be looked at in detail. Quickly determining optimal exploration parameters to help analyst interactively explore visualization and effectively find dataset characteristics becomes a necessity for all kinds of trajectory analysis tasks.

In this paper, we present a viewpoint-based trajectory visual exploration approach which integrates the novel 3D trajectory visualization techniques and viewpoint generation algorithms to avoid the above issues. Given the input data and application requirements, our approach can automatically generate a 3D trajectory visualization together with a viewpoint sequence that provides the analysts an optimized exploration manner to quickly grasp the overview characteristic of the generated visualization. To avoid the uncertainty of the viewpoint generation algorithm caused by the diversity of the trajectory datasets, we propose a generic viewpoint generation framework with a client-server architecture. The analyst first interactively select a route on a software interface (see Section 6.1), then all the trajectories containing the selected route are collected to construct a visualization and the corresponding viewpoints are generated as well. The generated viewpoints provide optimal perspectives to the visual exploration of the generated visualization to support the viewer to sequentially and effectively observe different levels of the visualization. Different from existing 3D trajectory visualization techniques that mainly focus on the visual design, our framework attempts to balance the visual expression and human cognitive effects.

In summary, the major contributions of this paper include:

- A visual exploration framework for quickly exploring trajectory data based on viewpoint.
- Novel 3D visualization techniques for displaying attribute distribution of trajectory datasets.
- Two types of viewpoint generation algorithms specialized in observing the details and overview information respectively.
- A systematic evaluation, consisting of a case study with a real bus dataset and a usability experiment.

The remaining part of this paper is organized as follows. Section 2 reviews related work. The approach overview is described in Section 3, followed by the visual design, viewpoint selection algorithms and concrete usage of our approach in Sections 4–6 respectively. Section 7 evaluates our approach in a case study and an experiment. Finally, we discuss the advantages and drawbacks of our approach in Section 8 and conclude the paper in Section 9.

2. Related work

2.1. Trajectory visualization

As the most common type of spatiotemporal data, trajectory data are analyzed in numerous visualization works, involving multiple visualization strategies:

Multi-view is one of the most common used strategies of trajectory visualization. Guo et al. [8] developed a trajectory analysis system consisting of a radial layout map, a ThemeRiver, a Scatterplot, a Parallel Coordinate, etc. to analyze the traffic trajectory data at a road intersection. Nagel et al. [9] developed a multi-view system for visually exploring public transportation trajectory datasets on a multi-touch tabletop. Liu et al. [10] designed a radial layout visualization to find different routes connecting two regions. However, these works simultaneously contain several equally important views to show different facets of trajectory, lacking intuitive and compact methods for showing the overview in a single view. Operating a multi-view system mainly depends on interaction, which affects the analyst on quickly understanding the dataset and discovering the potential knowledge from huge amounts of trajectories.

Utilizing aggregation and clustering to reduce the complexity of trajectory is another common strategy. Andrienko et al. [11–13] designed a series of trajectory visualization techniques based on spatiotemporal aggregation and clustering. They also proposed a technique taxonomy for modeling the relationship between facets of trajectories and analysis tasks [14]. Landesberger et al. [15] aggregated huge amounts of trajectories to form a graph, in which node and edge represent spatial areas and trajectory transitions between two areas respectively. However, these techniques mainly utilize clustering and aggregation to reduce the complexity of facets to be visualized, lacking the capability of providing an overview of the whole dataset to help the analyst quickly identify high level spatiotemporal characteristics.

Coordinate transformation, or converting absolute locations (longitude/latitude) of moving objects to relative distances to a reference, is an effective strategy in analyzing group moving patterns. Crnovrsanin et al. [16] analyzed the trajectories of all persons at the scene of a terrorist explosion and identified multiple suspects by visualizing the distance variations of all the persons to different selected important locations, such as the exit or the explosion site. Andrienko et al. [17] found the difference of group movement patterns between human and baboon by visualizing the distance of each individual in a group to the real-time group center. However, coordinate transformation emphasizes on analyzing group movement patterns rather than providing intuitive explorative views.

With the effectiveness of 3D visualization on showing spatiotemporal data and the global view of dataset [5], an increasing number of 3D trajectory visualization approaches have emerged. Tominski et al. [1,18] proposed a stack-based trajectory wall method for exploring multiple trajectories on the same route in a 3D context. Two similar methods named Great Wall of Space-Time [2] and Space-Time Cube [19] have been proposed by different researchers. Tominski et al. [20] put 3D icons on a map to visualize the time dependent data, and Rush et al. [9] proposed Space-Time Cloud to explore the public transportation data. Although these works support the exploration of the space, time and attribute information of multiple trajectories in one view, they do not resolve the inherent problems in 3D visualizations, such as clutter, overlapping, angle of the sight line affecting the display effects, and slow interaction speed, as shown in Fig. 1.

In this paper, we propose a comprehensive trajectory visualization framework. Our work integrates novel trajectory visualization techniques and viewpoint selection algorithms that guide the analyst to quickly explore the visualization space and to extract useful information from the analyzed datasets.

2.2. Visualization on Google Earth

In the last decades, proliferating research results have been made in the visualization and visual analysis of all kinds of data on Google Earth. Munro and Siemens [21] and Slingsby et al. [22] visualized the traffic state and insurance data on Google Earth. However, the visual designs of the above two works are simple, using the basic elements, such as icon, polyline, and polygon to represent the geographic dimension of the objects. Four new information visualization techniques on Google Earth contained in a spatial-temporal data exploration system were proposed by Wood et al. [23,24], which can be viewed as an attempt of integrating information visualization techniques into Google Earth. However, such techniques are 2D, not fully utilizing the 3D spatial representation capability of the platform. Sun et al. [25] developed a web-based visualization platform for climate research, and Chen et al. [26] used Google Earth to visualize A-Train vertical profiles. The advantages and limitations of using Google Earth as a virtual globe tool for earth science application have been reported by Yu and Gong [27]. To ensure the usability and learnability, we use Google Earth as the visualization platform. All the generated visualizations and viewpoints are recorded in the form of KML (Keyhole Markup Language) [28]. KML is an extension of XML for descripting of all kinds of geographic elements and widely supported by most of the GIS platforms. Google Earth is not only a GIS platform for rendering 3D spatiotemporal data, but more importantly provides huge volumes of freely available satellite images containing the contextual information, and an open standard for sharing the research findings and the information among users.

2.3. Viewpoint and interaction

Solutions to viewpoint selection have been proposed for understanding large and complex 3D objects, because automatically guiding users to good viewpoints improves both the speed and efficiency of data browsing. Vazquez et al. [29] defined the viewpoint entropy from the projected areas of the faces of the geometric models to derive good viewpoints. Tao et al. [30] proposed a unified framework, which solves the streamline selection and viewpoint selection by constructing two interrelated information channels between a set of streamlines and a set of viewpoints. Ji and Shen [31] proposed a time-varying viewpoints generation method. Bordoloi and Shen [32] presented a viewpoint selection algorithm for volume rendering, using voxel-based entropy as the evaluation function. However, most of existing viewpoint generation methods belong to the scientific visualization domain. To our knowledge, no information visualization or visual analytics work has utilized viewpoint-related techniques to improve the cognition effects of visualizations.

Several works conducted on interaction techniques have the same objective as the viewpoint selection. Qu et al. [33] proposed a Focus+Context zooming algorithm in 3D urban environments to help the viewer to quickly observe the buildings and roads of interest. Hurter et al. [34] designed an interactive trajectory visualization tool, named FromDaDy, to select trajectories from millions of records. A related design concept is that of Prezi [35]–a slice making software that first constructs an overview of all the slices, then generates a series of viewpoints, each zooming in a local part of the overview. These viewpoints are switched according to a predefined order for better understanding the overall knowledge structure. Inspired by these techniques, our visual exploration framework provides two types of viewpoints, namely overview viewpoint and detail viewpoint. After achieving a global optimal exploration at an overview viewpoint, the generated visualization volume can be interactively decomposed into a set of components according to analysis requirements, each having a detail viewpoint for better exploring local details. Furthermore, in addition to free navigation in visualization space, our framework also supports to predefine an exploration route consisting of multiple viewpoints to improve exploration efficiency.

Although several related studies have been conducted on viewpoint and interaction to improve cognition effect, little effort has been devoted to the visual exploration and analysis of trajectory. There is a need for such an approach for quickly exploring and understanding moving patterns of huge amount of trajectories in a 3D visual analytics context, which is the main motivation of our work.

3. Approach overview

3.1. Data and tasks

Trajectory dataset D having multiple facets can be formally defined as follows. A trajectory $t \in D$ is an ordered set of route points $t = (p_1, \ldots\ldots, p_n)$. Each route point p_i: $1 \le i \le n$ belongs to a high-dimensional space $p_i \in (S \times T \times A_1 \times \ \ldots\ldots \ \times A_m)$, where S, T, and $A_1, \ldots\ldots, A_m$ represent space component, time component and multiple attribute components respectively.

The general goal of trajectory exploration is to understand the interrelations between the different components, i.e. space S, time T, and attribute A, in trajectories of moving objects [13]. There exist different types of trajectory datasets, such as public transportation datasets having fixed routes and taxi datasets in which each trajectory has a distinctive spatiotemporal range. However, most of the visual analytics tasks, such as identifying traffic jam sections, consider interactively selecting a route and visualizing the attribute distribution of each individual route as a basic visual exploration function [1,3,8,36]. This point is crucial for our approach.

This paper proposes an approach that supports the analyst to interactively select a route and to quickly and effectively complete the general task based on predefined viewpoints. The analyst can quickly arrive at a series of optimal viewpoints, which expose most parts of the generated trajectory visualization to the analyst with optimal observation manners.

3.2. Framework

To treat the visualization and the viewpoint selection as an integral problem and offer a unified solution, our framework divides the exploration process of trajectory datasets into two phrases — visualization/viewpoint generation and visual exploration, as in

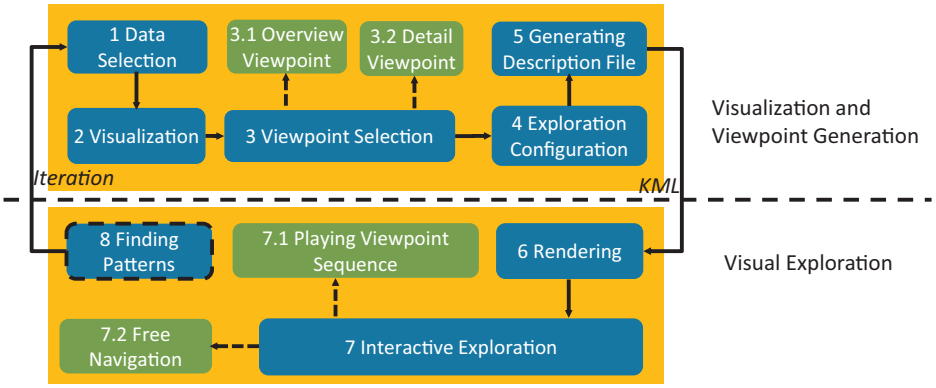


Fig. 2. The viewpoint-based framework for the visual exploration of trajectory data.

Fig. 2. In the first phrase, the trajectory datasets are prepared for exploration. The basic idea is to create a visualization and a series of viewpoints. The analyst first filters the dataset and attributes (Step 1), and selects appropriate techniques to visualize the data (Step 2). Then, we determine the viewpoints for the user to better explore and understand the visualization (Step 3), which can be either automatically generated by our viewpoint selection algorithms or manual setting. We propose two types of viewpoints, specialized in overview (Step 3.1) and details (Step 3.2) respectively. To better understand a view, the two types of viewpoints could be jointly used according to the analysis requirements. These viewpoints form an observing sequence. The analyst can proactively control the exploration process by predefining the order of visiting the viewpoints and the residence time at each viewpoint (Step 4). The generated visualizations and viewpoints are stored in a description file used in the next visual exploration phrase (Step 5). To facilitate the next exploration process, the description files are in the form of KML, which have been supported in most GIS platforms.

Having obtained the description file, the user can utilize any off-the-shelf GIS platform to render the visualization (Step 6). Basing on the toolkits of the GIS platform, our approach provides two types of exploration manners. The analyst can either continually explore the visualization according to the predefined exploration route by directly playing the viewpoint sequence in the GIS platform (Step 7.1) or manually control the screen to freely navigate in the visualization space (Step 7.2). The framework supports an iterative hypothesis-investigation process: the patterns found (Step 8) in the second phrase can also be used for the next round of investigation to select the appropriate dataset in the first phrase.

Separating computation-intensive tasks and visualization-oriented tasks can balance the loads of the whole exploration process. Furthermore, the framework also encourages sharing the analysis results among different types of users.

4. Visual design

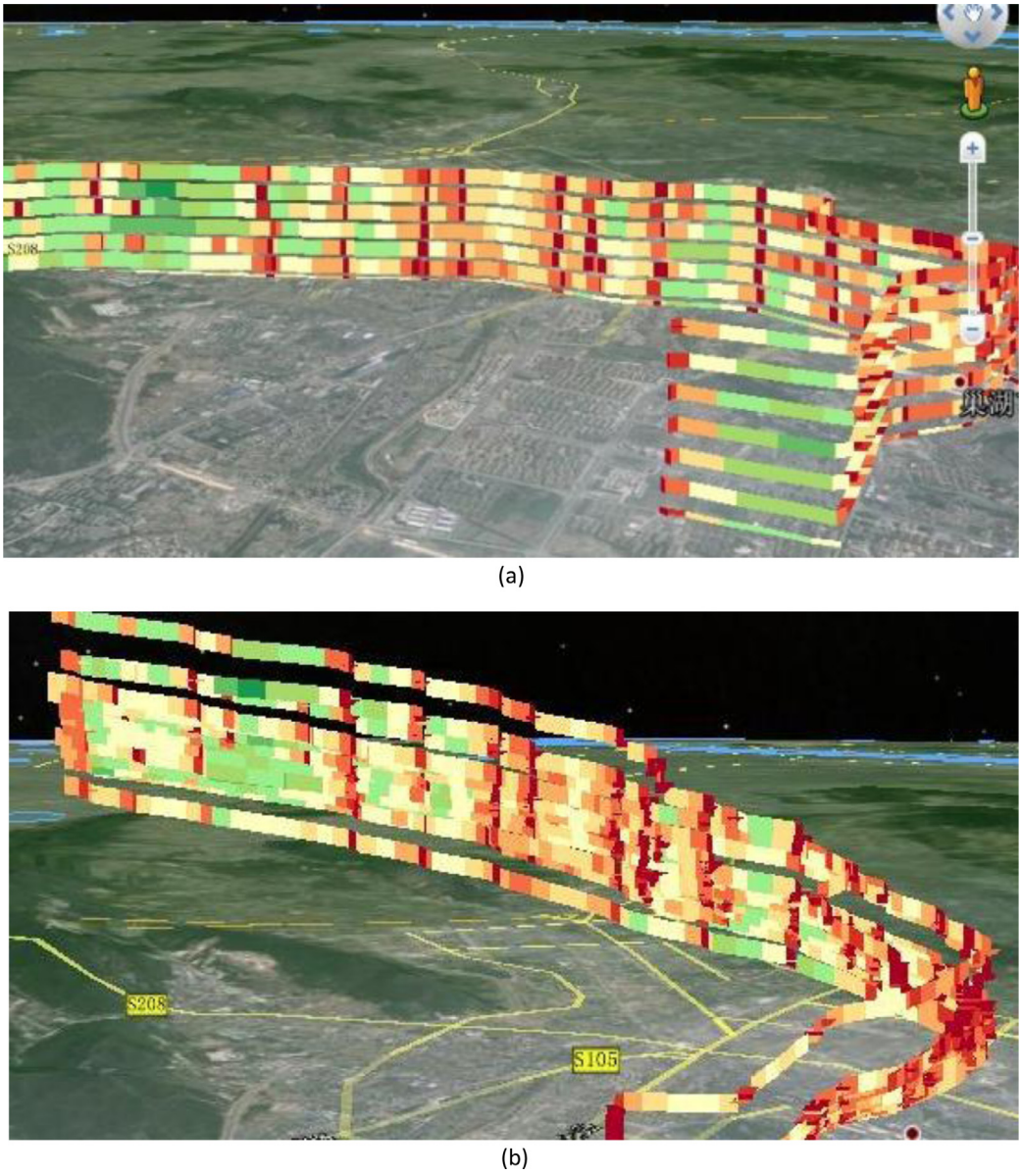
In order to visualize the attribute distribution, it is necessary to show trajectory attribute values in their spatiotemporal context. Since it is difficult for 2D visualization techniques to separate individual trajectories and discern attribute values along trajectory paths, we choose a 3D solution that utilizes the third dimension to stack multiple trajectories having the same route as a wall. The manner that stacks multiple trajectories in a 3D geographic context and color-codes the attributes, has been proved to be an effective method for simultaneously showing multiple facets of trajectory data [1,2]. The basic trajectory wall is shown in Fig. 1b. To show the temporal facets, the trajectories having similar spatial distributions are stacked from bottom to up in the chronological order.

Because we use the spatial coordinates in the visualization space for showing the dimensions of time and space, attribute values must be encoded using another visual variables. We choose color because it is a widely accepted approach and fits well with the trajectory wall design. To make the attribute distribution easily detectable and interpretable, it requires appropriate mapping of the values to colors. We use ColorBrewer [37], which recommends multiple color schemes for different number of classes. The analyst can interactively choose a color scheme according to different domain-specific conventions. The definition of appropriate class intervals is intricate because there is no solution that can perfectly divide the value range of attribute into different classes. Therefore, our approach supports the divisions into equal intervals by quantiles according to statistical distribution by default, as well as providing the analyst an interactively selection mechanism of partition criteria.

Each segment of a trajectory is colored to represent a particular attribute. By viewing the color distribution on the wall, the user can easily understand the attribute distribution characteristics with respect to the space and time. To reduce visual loads and highlight important information, we utilize the color filtering [1], in which the prominence of the selected value range of the attribute is decreased (see Fig. 10b–e). Our prototype system (see Section 6.1) provides the user an interface to select several intervals to active the color filtering.

To better use the trajectory wall we need to address the issues in a 3D visualization environment. Overlapping is the first problem needed to be solved. Existing 3D visualization techniques often resolve this problem by improving the alpha of the color of the visualization and color filtering, which would cause adverse effects of unintended color blending. Different from the existing techniques, we make a gap between two adjacent trajectories (see Fig. 3a) along the same path, through which the user can observe the objects behind the wall. Although the wall consumes more vertical space, we could reduce this effect by appropriately setting the interval width. Interpolation algorithm also can be used in the trajectory drawing to improve the smoothness of the color variation between different route vertices.

Another problem with the original trajectory wall is that it cannot reflect the travelling time of each trajectory. The trajectories in a wall are sorted in the chronological order of the start time. The vertical axis of the trajectory wall only reflects the relative temporal sequence. A convenient solution to this problem is using the vertical axis to represent the absolute temporal scale instead of the relative chronological order. However, the trajectories will intersect if they simultaneously appear at a close location. In an extreme



(a)

(b)

Fig. 3. Improvements to the trajectory wall. (a) Setting an interval between two adjacent trajectories to observe the objects behind the wall. (b) Trajectory wall of rising style. The rising height represents the travelling time used in this section. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

case of traffic jam, multiple trajectories would potentially overlap at many locations of a route. Therefore, we retain the stacking layout to represent the relative order of all the trajectories, and encode the travelling time by raising the endpoint of each trajectory a distance according to the traveling time of this segment, as in Fig. 3b. Segments with larger slopes or intersecting with other segments should attract our attention, indicating abnormal travelling status. Furthermore, Google Earth also provides a temporal span selection component that supports time-related analysis tasks. The user can determine the styles of the trajectory wall interactively in our prototype system (see Section 6.1).

We also implement a parallel coordinate wall, as in Fig. 11a, by adding multiple axes along the route. The added axes make it convenient to show the attribute values of multiple trajectories at important locations. Because the parallel coordinate wall and trajectory wall have a similar spatial shape, all the viewpoint selection algorithms can be directly used in parallel coordinate wall.

5. Viewpoint selection algorithms

5.1. Definition

A viewpoint can be considered a camera and defined as a vector of 5 parameters (x, y, z, h, t). (x, y, z) forms the 3-dimensional spatial coordinate of the camera (longitude, latitude, altitude), while (h, t) indicates the *viewing direction*, i.e. the direction of line of sight (see the red solid line in Fig. 4). Assume p is a viewpoint, pc is the line of sight, w is a section on the trajectory visualization, and c is the center of w, then pc is orthographic if $pc \perp w$ (see the yellow solid line in Fig. 4). In our viewpoint model, (h) and (t) are defined as the angles between the vertical component and horizontal component of the actual line of sight (see the two red dash lines in Fig. 4) and pc, representing the horizontal yaw and the vertical pitch respectively, as in Fig. 4. We also define other two terms used in the following algorithms descriptions, i.e. *viewing distance* measuring the length of line of sight and *viewing range* indicating the maximal sector angle of the camera, as in Fig. 4.

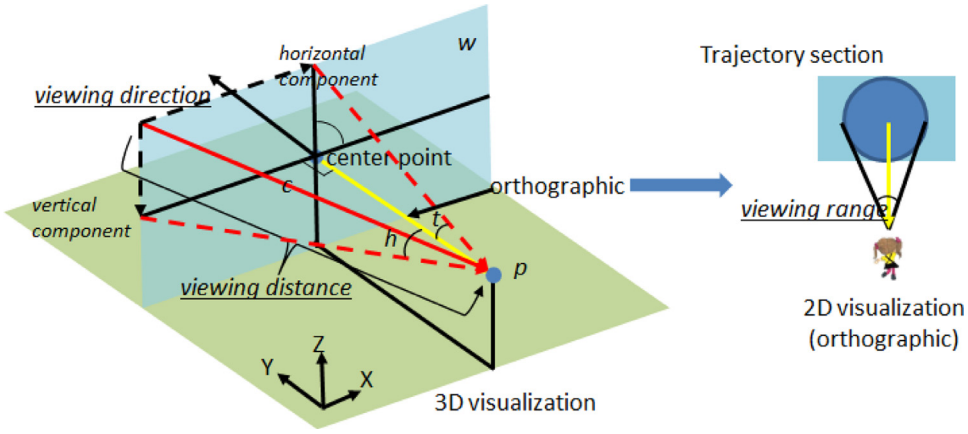


Fig. 4. Illustration of the viewpoint model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.2. Detail viewpoint

We define the viewpoint on the orthographic line of sight as orthographic viewpoint. Providing the best view of the wall section selected by a user, the orthographic viewpoint is therefore used for observing detailed information. Trajectory shapes may vary greatly, it is impossible to find a viewpoint suitable for observing the details of the entire wall. Most of the existing approaches require users to manually adjust the viewpoint. However, this requires more operation time in the 3D environment. Different from the existing approaches, our algorithm first divides the trajectory wall into multiple sections, each close to a straight line, and then establishes an orthographic viewpoint for each section. All the established viewpoints form a sequence that can be played in GIS platforms to explore the trajectory wall while maintaining the best viewing position. The algorithm is described in details below.

5.2.1. Dividing trajectory visualization

We first determine a set of points $S = \{s_1, s_2, \ldots s_N\}$ that divides the whole route consisting of multiple route points $v = \{v_1, v_2, \ldots v_M\}$ into N trajectory sections $w = \{w_1, w_2, \ldots w_N\}$. Each trajectory section contains multiple stacked trajectories on the same path. Our objective is to compute optimal camera parameters $p(x, y, z, h, t)$ for each section, as in Fig. 5. Given an angle threshold α and a distance threshold β, then a vertex s_i is added between v_i and v_{i+1}, if $angle\ (v_{i-1}, v_i, v_{i+1}) > \alpha$ or $length(v_i, v_{i+1}) > \beta$, in which the angle function indicates the turning *angle* between (v_{i-1}, v_i) and (v_i, v_{i+1}), and the *length* function represents the length of (v_i, v_{i+1}). In other words, if the direction of two adjacent lines changes a lot or the length of a line is too long to be covered by one viewpoint, a breakpoint will be added. Fig. 5 shows an example of trajectory being divided into 7 sections, in which s_1, s_2, s_3 and s_6 are added for a direction reason, while s_4 and s_5 are added for a distance reason. Seven viewpoints are added to different sections correspondingly.

5.2.2. Computing viewpoint parameters

Based on the viewpoint definition in Section 5.1, we compute the viewpoint parameters of each section. Let $c(\bar{x}, \bar{y}, \bar{z})$ be the coordinate of the center point of a section $s_i s_{i+1}$, the height of $s_i s_{i+1}$ is H, and $p_i(x, y, z, h, t)$ be the coordinate of the corresponding viewpoint, then p_i can be determined by resolving Eq. (1):

$$\begin{cases} y - \bar{y} = k'(x - \bar{x}) \\ (x - \bar{x})^2 + (y - \bar{y})^2 = r^2 \\ z = H/2 \end{cases} \quad (1)$$

In Eq. (1), k' is the slope of line pc, and r indicates the viewing distance. *The parameter* z of the viewpoint is always $H/2$ for being orthographic (see Fig. 5). For the rising trajectory wall (see Fig. 3b), it can be simply considered as the height of the midpoint of the corresponding section. Each section has two symmetrical detail viewpoints, which are introduced by the extraction of a root. To support a smooth transition between consecutive viewpoints, all the viewpoints take the same operation symbol (+ or -). For the rising trajectory wall (see Fig. 3b), z can be simply considered as the height of the midpoint of the corresponding

Viewing distance r is an importance parameter of Eq. (1). A reasonable value of r allows the section (or the intersection) to be viewed well on the 3D window of Google Earth. Because, setting the viewing distance depends on the viewing range of the window, we first have to determine the viewing range of Google Earth. We find that the initial viewing distance of Google Earth is 11,003.13 km, and the window can show the entire north-south diameter (12,630.824 km) in the initial view. Based on the above observation, we can obtain the viewing range of Google Earth, which is a constant and always equal to 60°, as in Fig. 6. Consequently, the viewing distance is set to $\sqrt{3}/2 \times \text{length}(w_i)$.

We also need to calculate other two direction parameters. For orthographic viewpoint, we can obtain h by computing the slope of pc to make the camera face the center point of the section. For orthographic detail viewpoint, t is always equal to 90°.

The computational complexities of obtaining the set of detail viewpoints P is O(M) (M is the number of the route points), satisfying the real time interaction requirements of the exploration process. The algorithm of computing the detail viewpoint is described as follow:

Algorithm 1. Determine the detail viewpoint.
Input: $\boldsymbol{v} = \{\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots \boldsymbol{v_M}\}$
$\boldsymbol{\alpha}$: ***angle threshold***; $\boldsymbol{\beta}$: ***distance threshold***
Output: $\boldsymbol{P}(\boldsymbol{p}, \boldsymbol{p_2}, \ldots \boldsymbol{p_N})$
$S = \{\}$
$P = \{\}$
For all $v_i v_{i+1}$ $0 < i < N$
if $angle(v_{i-1}, v_i, v_{i+1}) > \alpha$ *or* $length(v_i, v_{i+1}) > \beta$
$v_i \rightarrow S$
End for
Foreach s_i *in* S
compute parameters (x, y, z) *of* p_i *using Equation* (1)
compute parameter h *of* p_i *by making the camera face the center point of the section* w_i
set parameter $t = 90°$ *for orthographic reason*
$p_i \rightarrow P$
End for

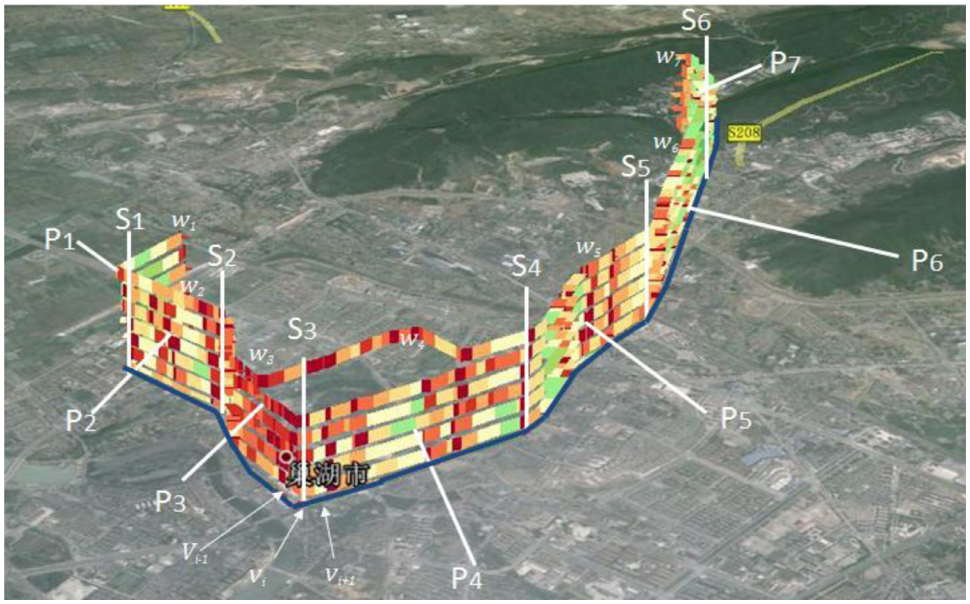


Fig. 5. Illustration of the viewpoint generation algorithm.

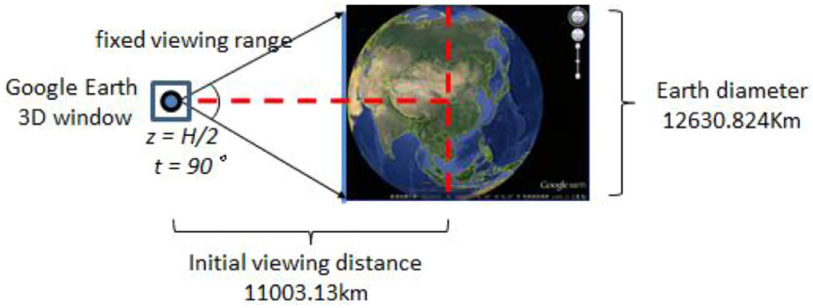


Fig. 6. Illustration of the determination of the viewing range of Google Earth. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.3. Overview viewpoint

Providing an overview is important in visual analysis of trajectory data. In most cases, it is impossible to show the entire trajectory wall due to the overlapping problem. According to the characteristics of trajectory data, we design a viewpoint generation algorithm for selecting a viewpoint that can expose the largest amount of information to the viewer. To view the majority of the wall, we attempt to find the *longest intersection* of the trajectory wall. Based on the definition of the spatial location set v (see Section 5.2), the *longest intersection* l can be described as follows:

$$l = \left\{ max\left(distance\left(v_i v_j\right)\right) | i, j \in (0, N) \right\} \quad (2)$$

The number of the overview viewpoints is not limited, and we can select multiple intersections for better understanding the trajectory data using Eq. (3), in which m indicates the number of viewpoints, and *top* function represents selecting top m length *intersections*.

$$l = \left\{ top\left(distance\left(v_i v_j\right), m\right) | i, j \in (0, N) \right\} \quad (3)$$

It can be easily inferred that one can see the longest wall when facing the longest intersection l. Therefore, overview viewpoint o must be on the line that is perpendicular to l and passes through the midpoint m of the longest intersection, and the parameter h can be defined using the slope of the perpendicular of l. The spatial parameters x, y, and z of o can be determined after setting the viewing distance r, namely the length of line om, as in Fig. 7. According to the viewing range of Google Earth (see Fig. 6), we set $r = \sqrt{3}/2 \times \text{length}(l)$.

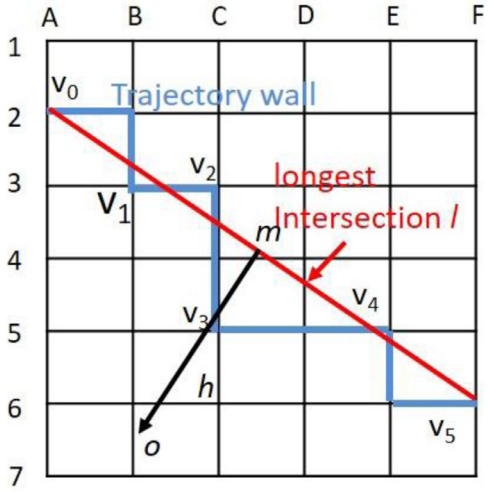


Fig. 7. Definition of the longest intersections.

We continue to calculate parameter t. To fully discuss the meaning of t for overview viewpoint, we consider two extreme cases: when $t = 90°$ (side view), the viewpoint is vertical to the wall, then attribute variation of the wall is clearly displayed, while

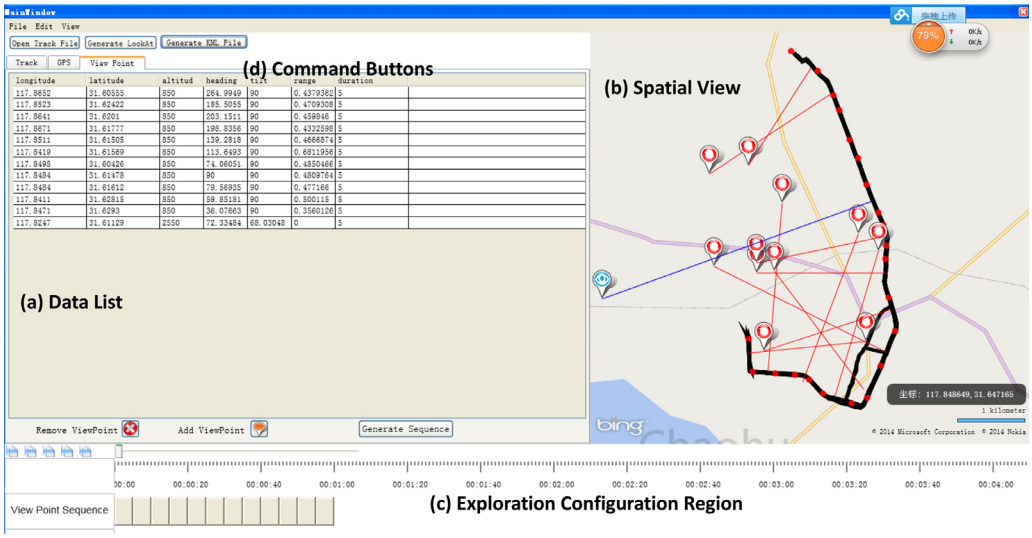


Fig. 8. Editing viewpoint using the prototype system. Blue and red marks represents overview viewpoint and detail viewpoints respectively, while the viewing direction is drawn as a directed straight line. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the spatial shape of the wall cannot be seen; instead, when $t = 0°$ (top view), we can only see the spatial shape of the wall, while the height of the wall is not shown. Therefore, we know that t is a variable that depends on whether we want to view the attribute variation or the spatial shape of the wall, and can be manually set according to the analysis task.

The computational complexities of the algorithm is $O(N^2)$, since we have to iteratively compare the distances between any two route points to obtain the longest intersection. Due to the fact that the numbers of route points of most trajectory sections are not less than 10^4, our algorithm can output the overview viewpoint in real time.

6. Usage of the approach

6.1. Prototype system

Our approach has a client-server architecture, where the server is written in C# and implements all the steps in the first phrase of the framework (see Section 3.2). All the analyzed dataset should be first imported into the system and shown in a *Data List*. By using the interface controls in *Data List*, the analyst can conveniently filter data according to attribute values.

When selecting a trajectory record in Data List, all the filtered records having the same spatial route are automatically collected to generate the visualization. Furthermore, the analyst can interactively click multiple coordinates on the *Spatial View* to form a polyline route. For each selected route, the system can automatically generate a viewpoint sequence consisting of an overview viewpoint and multiple detail viewpoints. Both the viewpoint sequence and the selected route are shown in the *Spatial View*.

Our system also supports to interactively add, delete and modify the locations of the generated viewpoints, as in Fig. 8, through interactive map operations. Each generated viewpoint is shown as a gray rectangle in the *Exploration Configuration Region*. The length of the rectangle indicates the residence time at the viewpoint during the exploration process. Our system supports the analyst to proactively edit the orders of accessing viewpoints and the residence time at each viewpoint through dragging and stretching a rectangle.

Three *Command Buttons* on the top of the interface are used to control the above workflow. For each exploration process, the analyst sequentially clicks a *Command Button* to complete several steps of the exploration framework (See Section 3.2).

Considering the diversity of the trajectories, it is a necessity to accommodate different types of trajectory datasets. For the trajectory datasets having fixed routes, such as bus trajectories, we can directly select a route as the exploration target. Furthermore, the system also provides a trajectory aggregation function [15] that enable the analyst to interactively define a route by clicking multiple coordinates on the *Spatial View*. Through aggregating the trajectory sections containing any pair of coordinates, we can obtain a single trajectory that has the same spatial shape as the defined route. Furthermore, we can group all the trajectories according to time and generate a route for each group to represent the movement situation of the route at a specific time interval.

It should be noted that our approach does not limit the number of the selected routes. The analyst can simultaneously choose multiple routes for the comparison purpose. All the selected routes and the corresponding viewpoints are stored in a KML file. During the procedure of visual exploration, the analyst can control the visibility of each route using the interaction toolkits of the GIS platform, and separately explore the attribute distribution of each selected route.

6.2. Visualization platform

Different from most of the 3D visualization techniques are constructed in customized platforms, our approach is implemented on KML supported within the Google Earth API. The standard I/O interface and visualization description mechanism ensure the usability and learnability and benefit sharing the visualizations and the analysis results among users.

The imported trajectory sections and the viewpoint sequence are listed in the left panel of Google Earth. When clicking on a trajectory section, the screen is switched to the corresponding detail viewpoint to explore the selected section, while clicking on the viewpoint sequence triggers the autoplay process. The user can alternatively use viewpoint-based manner and free navigation according to analysis tasks. Furthermore, Google Earth provides detailed geographic contextual information, such as terrain and cities' zone planning, which cannot be easily obtained from other sources and seamlessly integrated.

Fig. 9. Topview (Tilt = 0°) of the spatial distribution of all the routes in the datasets, each color indicating a route. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6.3. Descripting visualization and viewpoint using KML

We use the ⟨polyline⟩ tag instead of ⟨polygon⟩ tag due to color deviation and insufficient brightness caused by the illumination mechanism of Google Earth. This problem also prevents us from designing more effective visualizations. Because the number of coordinates used to construct a ⟨polygon⟩ is also less than ⟨polyline⟩, the size of the generated KML file is greatly reduced, which minimizes the CPU and memory usage for rendering the visualization. We utilize the ⟨look at⟩ tag to define the viewpoint model, and combines multiple viewpoints to form a viewpoint sequence by using the ⟨gx: Tour⟩ tag.

We also provide a mechanism for exploring the temporal variation characteristics. By adding a ⟨gx: time span⟩ tag to each section, we can define life cycles for different objects in the visualization space. A visual object can only be viewed from the viewpoints in its life circle. We can adjust the time interval need to be explored by using the temporal progress bar of Google Earth, through which the detail information of any subinterval can be exclusively rendered in the visualization space. Furthermore, we can explore the dynamical temporal variation of trajectory datasets by continuously dragging the temporal progress bar.

7. Evaluation

7.1. Case study

We use an actual bus trajectory dataset to evaluate the effectiveness of our approach. The dataset contains about 10 million bus GPS records of Chaohu (a small city in China's Anhui province). The public transportation system in Chaohu consists of 23 routes and 272 bus stations, as in Fig. 9.

In order to verify the effectiveness, our approach visualized 20 tracks during 10:00 to 20:00 having relatively complete GPS records in route 1. To highlight the congested sections, we used the color filtering style [1] that reduced the wall width of the section having a faster speed, making us more focus on the congested section. The entire wall was divided into 11 sections according to its spatial shape, each having a detail viewpoint to the corresponding section. The algorithm of detail viewpoint sets $\alpha = 15^\circ$ and $\beta = 10$ km.

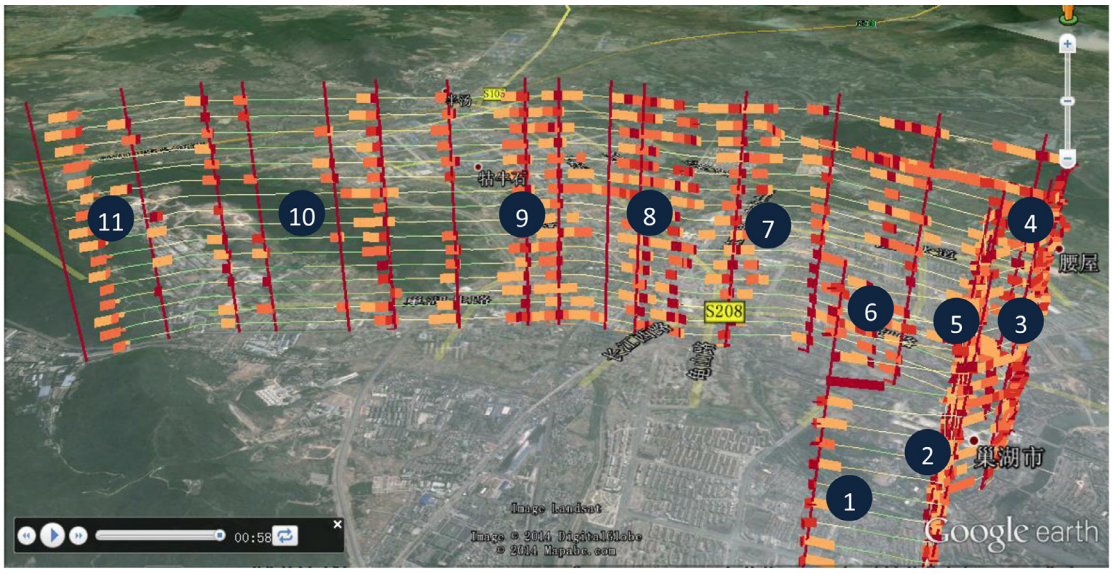
An overview viewpoint (see Fig. 10a) was also generated, which sets $tilt = 45°$to observe both the attribute variations and spatial shapes of the trajectories. The positions of all the 12 viewpoints are shown in Fig. 8. The duration of each detail viewpoint was 2 s, while the overview viewpoint had a longer observation time of 5 s. All the viewpoints formed an exploration route, and the overview viewpoint was set at the beginning. The visualization and the viewpoints were finally exported to a KML file. By exploring the visualization from an overview viewpoint, as in Fig. 10a, we quickly found that most of the places having slower speeds were near bus stations (the red vertical line in Fig. 10a–e) except the section in the 4th viewpoint (see Fig. 10c). Data analysts explained that the collected dataset originated from a small city, in which the traffic jam problem was not series and the found section having lower speed was located at a business district (see Fig. 11c). We also found that the trajectories at the top of the section in the 8th viewpoint also had relative slow speeds (see Fig. 10d). Because the time intervals of these trajectories are between 17:00 to 19:00, the slow speed distribution obviously indicated an evening rush hour.

The prototype system also implements the parallel coordinate wall. We used this style to analyze the number of passengers getting on the bus at different stations. The coordinate axes were set at the bus stops, and the height of trajectory indicated the number of passengers. Totally 5 trajectories between 16:00 and 18:00 of Route 1 are selected. From the overview viewpoint (see Fig. 11a), we found the wall was particularly high in two areas. We manually operated Google Earth to obtain the detailed satellite images of the two areas. Having zoomed in, we found a playground in each area, apparently indicating a school. Furthermore, according to our priori-knowledge, the second area is a business district, which can also account for the found patterns. Our finding is consistent with the real situation in Chaohu, where the usage of bus cards is less than 20% and only students or employees living far away use it.

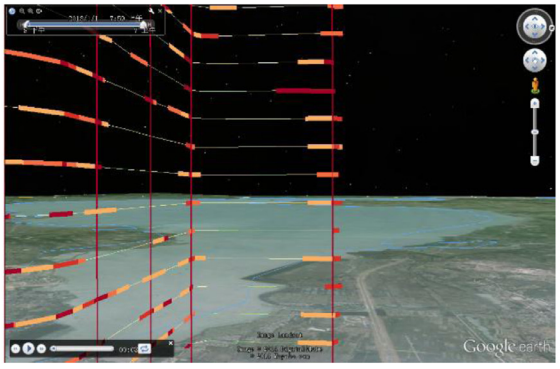
7.2. Experiment

7.2.1. Objectives and hypotheses

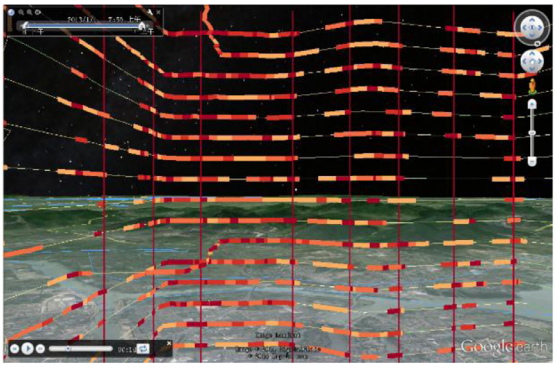
The goal of the experiment is to evaluate the effectiveness of the viewpoint-based visual exploration approach. Since the functionality and correctness of the approach have been verified through the case study, the experiment focuses on whether our approach can significantly improve the cognition effects of the exploration process. More specifically, we hope to quantitatively compare our approach with the original trajectory wall technique without predefined viewpoints. According to the aforementioned objectives, we made two hypotheses for the following experiment.



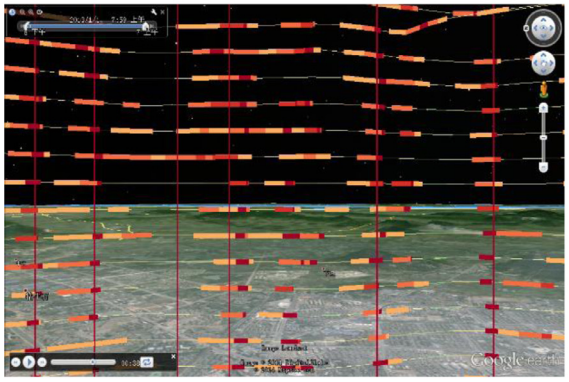
(a) Overview viewpoint (Tilt = 45°)



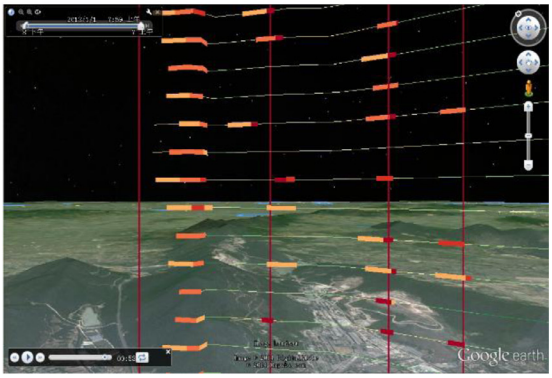
(b) Detail viewpoint 1 (Tilt = 90°)



(c) Detail viewpoint 4 (Tilt = 90°)



(d) Detail viewpoint 8 (Tilt = 90°)



(e) Detail viewpoint 11 (Tilt = 90°)

Fig. 10. Exploration of 20 trajectories in Route 1 using our approach. (b) A side view with the overview viewpoint, marked with 11 detail viewpoints. (c–e) Two detail viewpoints. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- *Hypothesis* 1: Our approach has a shorter *completion time* than the original trajectory wall without predefined viewpoints.
- *Hypothesis* 2: The viewpoint-based approach can significantly improve the *correctness* of the trials.

7.2.2. Subjects

For the study, we recruited 32 participates from the School of Software Engineering, Tianjin University as experiment subjects. Two of the subjects were female, while others were male. All the subjects were graduate students (aged 23–30), of whom 10 had experiences in visualization, and 4 had analyzed the public transportation data. All the subjects were confident with mouse and keyboard interaction, and 15 of them had used Google Earth. None of the subjects had used our approach.

7.2.3. Tasks

We chose a between-subjects study design to eliminate the learning effect, and the subjects were randomly divided into two

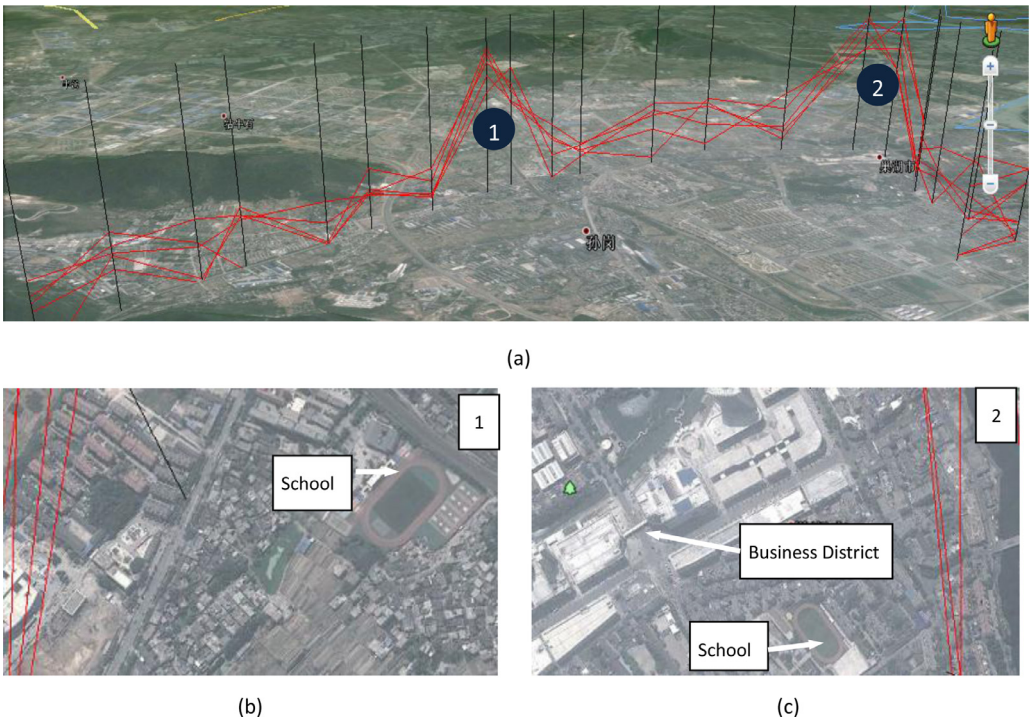


Fig. 11. A parallel coordinate wall consisting of 5 trajectories. (a) Overview viewpoint with $tilt = 60°$. Two bus stops have bigger values, indicating that more passengers get on bus at the two bus stops. (b–c) A zoomed in view of the two marked areas in (a).

equal groups, which performed the experiment using our approach and the original trajectory wall respectively. We designed 5 tasks that comprehensively covered all the elements, each deriving from an actual analysis task:

- Operating the visualization and locating a specified route section.
- Finding the congested section of a route.
- Finding the station has the most passengers.
- Finding the most congested area during a specified temporal interval.
- Estimating the average travelling time of a route.

Without loss of generality, each task was conducted on two routes, which results in a total of 10 trials for each subject. The *completion time* and the *correctness* of each trial were manually recorded when performing the experiment. To ensure the consistency of the time records, the initial status of each trial was set to the opening view of Google Earth. All the trials were conducted on the same laptop with a screen resolution of 1440*900 pixels in a quiet laboratory environment.

7.2.4. Stimuli

Ten KML files consisting the visualization and the corresponding viewpoints had been created before the experiments, each for one trial. Each KML file contains an overview viewpoint and multiple detail viewpoints, and the overview viewpoints were set as the first frame of the viewpoint sequences by default. All the routes and temporal intervals were carefully selected by containing congested sections to make the experiment results significant. Having understanding the general shape of the visualization, each subject could interactively select a viewpoint from the viewpoint list on the left of Google Earth interface according to the task and explore the corresponding subarea. Two groups used the same KML files. Group A used the visualization + viewpoint approach, while Group B only manually operated the visualizations.

7.2.5. Process

We first gave the subjects a brief introduction to the trajectory wall and viewpoints, and then spent 10–15 min training them on the usage of our approach in Google Earth. The training was divided into two phases. In the first phase, the subjects were free to view a trajectory using Google Earth to get familiar with the interface. In the second phase, they practiced playing a viewpoint sequence and switched to the frame that we asked them to view. We also answered the questions raised by the subjects to ensure the experimental environment to be functional and the subjects capable of completing the tasks. Any technical problems arose in the training were solved before the experiment started.

7.2.6. Result analysis

The experiment was a between-subjects study. Each session included 10 trials, taking 4–6 min. We chose one-way ANOVA with two dependent variables, i.e. *completion time* and *correctness*, and one factor, i.e. *group*. Before the analyses of variance (ANOVA), data were routinely checked for variance homogeneity of the selected dependent variables. Although the *correctness* did not satisfy the variance homogeneity, we continued to verify the two predefined hypotheses considering the relative small sample size:

Hypothesis 1: The difference of the *completion time* between the two groups is significant ($F = 19.903,\ P = 0.01$), indicating our approach has a faster interaction speed than the original trajectory wall. In our approach, the subjects can quickly find the targets by viewing the overview viewpoint, and switch to the targets by clicking the corresponding detail viewpoint in the viewpoint list on the left of Google Earth graphic interface. On the contrary, subjects had to manually control the viewpoint, when using the original trajectory wall. Sometimes, although the subjects had already found the target area, they had to manually adjust the viewpoint to that area to determine if the findings were correct. For the tasks that need continuous viewing of the details of multiple sections in one route, such as Tasks 1 and 2, the advantages of our approach are more obvious, while for the tasks that focus on the overview, such as

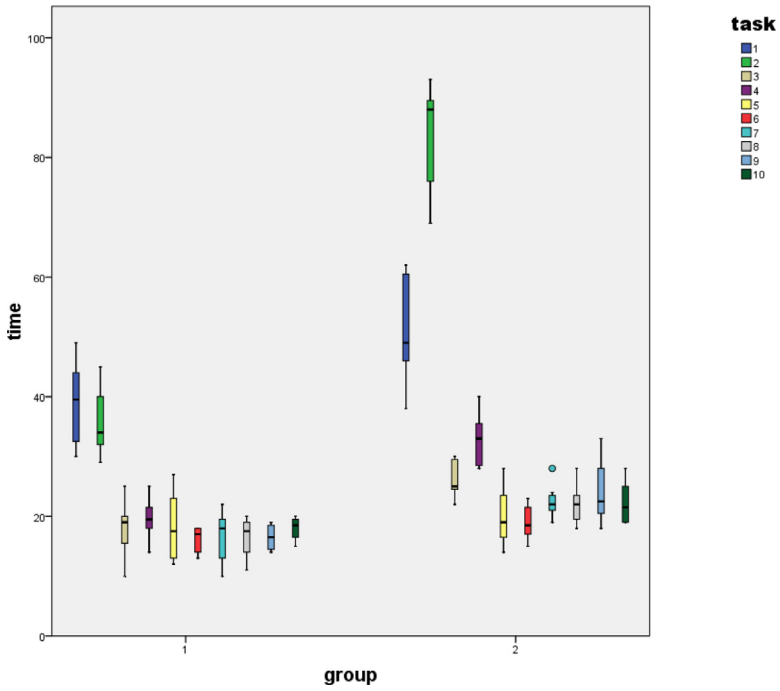


Fig. 12. The boxplot of the experiment results, each color indicates a task, from which we find the differences of *completion time* of tasks 1 and 2 between two groups are more significant than other 8 tasks. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Tasks 5, 6, 7 and 8, the differences of the time taken are insignificant, as in Fig. 12. This result is caused by the fact that large percentage of completion time takes place in the phase of viewpoint change. If the tasks need to frequently switch viewpoints, our approach can save much time.

Hypothesis 2: The statistic result did not show a significant difference on *correctness* ($F = 1.704,\ P = 0.194$) between the two groups. The average correctness of the two groups are 91.25% and 96.25% respectively, which indicates a ceiling effect. The routes in the experiment are carefully selected so that the characteristics can be easily recognized and the subjects have unlimited experimental time, which makes them spend too much time on verifying the findings.

8. Discussion

Our experiments have shown that the viewpoint-based exploration approach has better performance than the original Trajectory Wall. One of the most important characteristics of the approach is to help the analyst quickly move to a series of optimal viewpoints, which significantly improves the exploration effect and efficiency of the exploration. In addition, we improved the usability and scalability of our approach in two aspects. First, all the visualization and viewpoints are described by Keyhole Markup Language (KML), which is an international standard for sharing geographic information. Second, we utilized Google Earth as the visualization platform. The interaction tools and the contextual information contained in the satellite images of Google Earth make our approach adaptable to lots of analysis tasks of different types of trajectory datasets. Next, we demonstrate how the approach can improve the exploration effect and efficiency through a case study and a usability experiment. Both the discovered patterns and the experiment results proved the effectiveness of our approach.

Generally, the exploration manner similar to the physical world received very positive feedback from the subjects of the experiment. All the subjects can quickly understand the visualization and smoothly use the components of Google Earth to perform all kinds of analysis tasks. The function that allows the user to quickly switch between multiple predefined viewpoint series is beyond their expectations, and they comment that our approach is simple and effective.

We found that the subjects preferred to use the time axis control of Google Earth to do the time-related tasks rather than using the trajectory wall of rising style. This implies that the visual design of our approach needs to be improved. Two subjects pointed out that the trajectory wall would be too high to observe, if huge amounts of trajectories were visualized. However, this problem can be potentially resolved by setting a rational length of the interval. Most subjects considered the operations on Google Earth were not smooth and response time was relatively long when more than 30 trajectories were rendered in Google Earth, which affected the performance aspect of our approach. This is mainly because the experiment was conducted on a laptop. Using a high performance computer could resolve this problem. The powerful features of Google Earth in obtaining the geographic and contextual information from its satellites received very positive comments. The analyst can associate the found patterns with the actual contextual environment, which largely enhances the analysis capability of our approach in trajectory data exploration. In comparison with the original visualization, our approach makes the exploration process smoother and more enjoyable.

The subjects also pointed out a limitation of our approach: “trajectory is a diverse data structure, and there exists several types of trajectory dataset, although our approach is applicable to explore the spatiotemporal attribute attribution of a single route, we cannot guarantee it can be properly in other scenarios”. However, trajectory simultaneously has multiple facets, and it is impossible to show all of them in one view. A visualization needs to utilize all the visual elements to encode the facets related to the core analysis requirements. Since displaying the attribute distribution of a single route is one of the most popular tasks in trajectory analysis, our work solves a common problem in 3D trajectory exploration and can be seamlessly integrated with other existing analysis systems of trajectory datasets. Furthermore, if users wish to explore

trajectory datasets that have arbitrary trajectory shapes, the trajectory aggregation function of the prototype system can be exploited to determine a route by directly clicking on the map. We therefore hope that our approach provides an effective manner for experts to interact with 3D visualization and guide them for more flexible and in-depth studies based on the capability of quickly and optimally navigating in a 3D visualization space.

9. Conclusion

This paper has presented a viewpoint based approach for exploring trajectory data. New techniques for visualizing trajectory data, as well as two types of viewpoints specialized in different exploration purposes, have been developed and integrated into our approach. Comparing with other trajectory exploration strategies, our approach assigns equal importance to visual design and exploration effect, and resolves the inherent problems in 3D visualizations by predefining a viewpoint sequence. Furthermore, all the generated visualization and the viewpoints are described in the form of KML and can be rendered in Google Earth, which is convenient for the analyst to directly utilize the components of Google Earth to explore the visualization and to integrate actual contextual images into analysis tasks. A systematical evaluation consisting of a case study and a usability experiment has been conducted on an actual bus trajectory dataset, through which we consider our approach to be effective and useful in real-world scenarios. In the future, we plan to improve the approach in two aspects. First, we will improve the efficiency of the trajectory aggregation function to make our approach better accommodate trajectory datasets with different spatiotemporal characteristics. Second, we will attempt to enrich the visual design of 3D trajectory visualization in the Google Earth.

Acknowledgments

We are grateful to the anonymous reviewers for their insightful comments that have helped us in improving the final presentation. The work is partially supported by NationalNSFC project (Grant number 61602340), National NSFC project (Grant number 61572348) and National High-tech R&D Program (863 Grant number 215AA020506).

References

[1] C. Tominski, H. Schumann, G. Andrienko, N. Andrienko, Stacking-based visualization of trajectory attribute data, IEEE Trans. Vis. Comput. Graph. 18 (12) (2012) 2565–2574.
[2] C. Tominski, H.J. Schulz, The great wall of space-time, in: Proceedings of Vision, Modeling & Visualization, 2012, pp. 199–206.
[3] N. Andrienko, G. Andrienko, S. Wrobel, Visual analytics of movement: an overview of methods, tools and procedures, Inf. Visual. 12 (1) (2013) 3–24.
[4] A. Kjellin, L.W. Pettersson, S. Seipel, M. Lind, Evaluating 2D and 3D visualizations of spatiotemporal information, ACM Trans. Appl. Percept. 7 (3) (2010) Article 19.
[5] E. Kaya, M.T. Eren, C. Doger, S. Balcisoy, Do 3D visualizations fail? An empirical discussion on 2D and 3D representations of the spatio-temporal data, in: Proceedings of EURASIA GRAPHICS 2014, 2014 Paper 5.
[6] J. Rush, M. Kwan, Geovisualization of public transportation using the space–time cloud, in: Proceedings of the 25th International Cartographic Conference (ICC 2011), 2011 CO-421.
[7] T. Kapler, W. Wright, GeoTime information visualization, Inf. Visual. 4 (2) (2005) 136–146.
[8] H.Q. Guo, Z.C. Wang, B.W. Yu, H.J. Zhao, X.R. Yuan, TripVista: triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection, in: Proceedings of the IEEE PacificVis'11, 2011, pp. 163–170.
[9] T. Nagel, M. Maitan, E. Duval, A.V. Moere, J. Klerkx, K. Kloeckl, C. Ratti, Touching transport-a case study on visualizing metropolitan public transit on interactive tabletops, in: Proceedings of ACM AVI'14, 2014, pp. 281–288.
[10] H. Liu, Y. Gao, L. Lu, S. Liu, H.M. Qu, L.M. Ni, Visual analysis of route diversity, in: Proceedings of the IEEE VAST'11, 2011, pp. 171–180.
[11] G. Andrienko, N. Andrienko, Spatio-temporal aggregation for visual analysis of movements, in: Proceedings of the IEEE VAST'08, 2008, pp. 51–58.
[12] G. Andrienko, N. Andrienko, C. Hurter, S. Rinzivillo, S. Wrobel, Scalable analysis of movement data for extracting and exploring significant place, IEEE Trans. Vis. Comput. Graph 9 (7) (2013) 1078–1094.
[13] G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, F. Giannotti, Interactive visual clustering of large collections of trajectories, in: Proceedings of the IEEE VAST'2009, 2009, pp. 3–10.
[14] G. Andrienko, N. Andrienko, P. Bak, D. Keim, S. Kisilevich, S. Wrobel, A conceptual framework and taxonomy of techniques for analyzing movement, J. Vis. Lang. Comput. 22 (3) (2011) 213–232.
[15] T. Von Landesberger, F. Brodkorb, P. Roskosch, N. Andrienko, G. Andrienko, A. Kerren, MobilityGraphs: visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering, IEEE Trans. Vis. Comput. Graph 22 (1) (2016) 11–20.
[16] T. Crnovrsanin, C. Muelder, C.D. Correa, K. Ma, Proximity-based visualization of movement trace data, in: Proceedings of IEEE the VAST 2009, 2009, pp. 11–18.
[17] N. Andriaenko, G. Andrienko, L. Barrett, M. Dostie, P. Henzi, Space transformation for understanding group movement, IEEE Trans. Vis. Comput. Graph 19 (12) (2013) 2169–2178.
[18] G. Andrienko, N. Andrienko, H. Schumann, C. Tominski, Visualization of trajectory attributes in space-time cube and trajectory wall, in: Cartography from Pole to Pole, Springer, Berlin Heidelberg, 2014, pp. 157–163.
[19] M.J. Kraak, The space-time cube revisited from a geovisualization perspective, in: Proceedings of the 21st International Cartographic Conference, 2003, pp. 1988–1996.
[20] C. Tominski, P. Schulze-Wollgast, H. Schumann, 3D information visualization for time dependent data on maps, in: Proceedings of the IEEE InforVis 2005, 2005, pp. 175–181.
[21] S. Munro, P.L.C. Siemens, Fine grained traffic state estimation and visualization, ICE Civil Eng. 167 (5) (2014).
[22] A. Slingsby, J. Dykes, J. Wood, M. Foote, M. Blom, The visual exploration of insurance data in Google Earth, in: Proceedings of the GIS Research UK 16th Annual Conference (GISRUK'08), 2008, pp. 24–32.
[23] J. Wood, J. Dykes, A. Slingsby, K. Clarke, Interactive visual exploration of a large spatio-temporal dataset: reflections on a geovisualization mashup, IEEE Trans. Vis. Comput. Graph 13 (6) (2007) 1176–1183.
[24] A. Slingsby, J. Dykes, J. Wood, Interactive tag maps and tag clouds for the multiscale exploration of large spatio-temporal datasets, in: Proceedings of IEEE InfoVis, 2007, pp. 497–504.
[25] X. Sun, S. Shen, G.G. Leptoukh, P.X. Wang, L.P. Di, M.Y. Lu, Development of a web-based visualization platform for climate research using Google Earth, Comput. Geosci. 47 (2012) 160–168.
[26] A. Chen, G. Leptoukh, S. Kempler, C. Lynnes, A. Savtchenko, D. Nadeau, J. Farley, Visualization of A-train vertical profiles using Google Earth, Comput. Geosci. 35 (2) (2009) 419–427.
[27] L. Yu, P. Gong, Google Earth as a virtual globe tool for Earth science applications at the global scale: progress and perspectives, Int. J. Remote Sens. 33 (12) (2012) 3966–3986.
[28] J. Rohlf, and B. McClendon. U.S. Patent No. 7,353,114. Washington, DC: U.S. Patent and Trademark Office, 2008.
[29] P.P. Vazquez, M. Feixas, M. Sbert, W. Heidrich, Viewpoint selection using viewpoint entropy, in: Proceedings of Vision, Modeling, and Visualization Conference, 2001, pp. 273–280.
[30] J. Tao, J. Ma, C. Wang, C.K. Shene, A unified approach to streamline selection and viewpoint selection for 3D flow visualization, IEEE Trans. Vis. Comput. Graph 19 (3) (2013) 393–406.
[31] G. Ji, H.-W. Shen, Dynamic view selection for time-varying volumes, IEEE Trans. Vis. Comput. Graph 12 (5) (2006) 1109–1116.
[32] U.D. Bordoloi, H.-W. Shen, View selection for volume rendering, in: Proceedings of IEEE Visualization, 2005, pp. 487–494.
[33] H.M. Qu, H. Wang, W. Cui, Y.C. Wu, M.Y. Chan, Focus+context route zooming and information overlay in 3D urban environments, IEEE Trans. Vis. Comput. Graph 15 (6) (2009) 1547–1554.
[34] C. Hurter, B. Tissoires, S. Conversy, Fromdady: spreading aircraft trajectories across views to support iterative queries, IEEE Trans. Vis. Comput. Graph 15 (6) (2009) 1017–1024.
[35] B. Perron, A. Stearns, A review of a presentation technology: prezi, Res. Social Work Pract. 21 (3) (2010) 376–377.
[36] N. Ferreira, J. Poco, H.T. Vo, J. Freire, C.T. Silva, Visual exploration of big spatio-temporal urban data: a study of New York City taxi trips, IEEE Trans. Vis. Comput. Graph 19 (12) (2013) 2149–2158.
[37] C.A. Brewer, M.A. Harrower, ColorBrewer 2.0: Color Advice for Cartography, Penn State University, 2009 Http://colorbrewer2.org/.