Multimedia Tools and Applications (2019) 78:35099–35118
https://doi.org/10.1007/s11042-019-08060-8

Expectation-based 3D edge bundling

Guibing Yang[1] · Kunle Ma[1] · Xiaohui Yuan[2] · Jie Li[1] · Qiang Lu[1]

Received: 22 March 2018 / Revised: 10 June 2019 / Accepted: 2 August 2019 /
Published online: 31 August 2019

Abstract
In the visualization of the node-link graph, it is common to use edge-bundling algorithms to reduce the visual clutter caused by the increase in nodes and connections while reflecting the high-level structure of the graph. However, the traditional force-directed edge-bundling method has unstable gravitation when applied in three dimensions. To address this issue, we propose an edge-bundling algorithm based on the expectation model, and the edge-bundling rules can be modularized to support the addition of calculation rules. The stability of the proposed method is improved. Our experimental results with 2D and 3D scenarios demonstrate that our algorithm produces superior results that unclutter complex graphs.

Keywords Edge bundling · Force-directed · Expectation model

1 Introduction

Interconnection data such as air traffic are usually abstracted with graphs [5]. Data visualization has been used as a tool for the analysis of airport construction, traffic optimization, and regional dependency. A meaningful and concise visualization of massive, complex data allows

✉ Qiang Lu
luqiang@hfut.edu.cn

Guibing Yang
spacemiao_yang@163.com

Kunle Ma
makunle@163.com

Xiaohui Yuan
xiaohui.yuan@unt.edu

Jie Li
lijie_62@163.com

[1] School of Computer and Information, Hefei University of Technology, Hefei, China

[2] Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA

users to derive information that could be obscure. As the volume of data grows, it becomes challenging to create a visualization that presents a coherent and uncluttered view. In addition, with an increasing number of nodes, edges, and dimensionality, readability of visualization could be significantly degraded.

To address the challenges from massive data of high complexity, edge-bundling algorithms have been developed. Edge-bundling uses a hierarchical structure to combine spatially adjacent edges [6] and extensions have emerged to accommodate bundling under various circumstances [3, 7, 8, 18], most of which aim at applications of the two-dimensional dataset. However, large graphs can be better appreciated when they are visualized in a three-dimensional (3D) space and displayed with stereo and/or motion-depth cues to support spatial perception [17]. With the increase of spatial-temporal, geographical data, methods that take advantage of the 3D visualization space is greatly beneficial to the end user of such data. For example, a visualization of air traffic in 2D is unable to visualize altitude differences between domestic and international flight routes. A projection of 3D routes to a 2D plane results in many crossing edges in a graph; information such as flight altitude, and route origin and destination can hardly be attained. 3D edge bundling extends 2D methods to achieve less clutter and greater differentiation. The force-directed edge bundling algorithm (FDEB) [7] adopts the simulated electrostatic force, graph bundling by kernel density estimation (KDE) [8]. The 3D mean-shift edge bundling (3DMSEB) [2] extended FDEB by updating the correlation calculation of compatible edges and the division of control points. However, methods such as FDEB faces challenges with the increased complexity of graphs [1]. It is also difficult to determine when the electrostatic and the spring forces change due to the growth of edges.

The 3D edge bundling algorithm is a method of bundling lines with nearby or content correlations in three dimensions. This method for solving data (such as flight data, including longitude, latitude, and altitude) that is covered by terrain and can cause visual clutter when directly rendering in 3D. One of the challenges of visualizing geographic data in 3D is extending the 2D edge-bundling algorithms to 3D. While there is a need to improve the algorithm to show more information in the data, it is also necessary to better combine the data and geographic information to make the visualization result natural and beautiful.

The force-directed edge bundling algorithm (FDEB) [7] is widely used as a reference approach because of its flexibility, usability, and extensibility. It adopts the simulated electrostatic force, graph bundling by kernel density estimation (KDE) [8]. For instance, 3D mean-shift edge bundling (3DMSEB) [2] refers to FDEB in the correlation calculation of compatible edges and the division of control points. Similarly, we choose to extend the FDEB algorithm in the process of edge bundling. During bundling, we experience problems with the length and find that the maximum length of the edge can reach 10^2 in 2D space. Therefore, the balance can be maintained by appropriately adjusting the spring constant k during the bundling process. In addition, a smaller step size allows control points to move gradually, which prevents long edges from being moved sufficiently, whereas a large step size causes severe fluctuations and destabilization. Moreover, it is difficult to determine the value of k when the electrostatic force and the spring force change due to the growth of edges. As stated by Bondi [1], FDEB does not perform well for large-scale graphs.

In this paper, we propose an edge bundling method based on the expectation model to address the problems from massive data of high complexity. Our method leverages expectation modules and addresses the instability of FDEB. A balance is achieved with an adjustable spring constant during the bundling process. The regulating step size moves control points gradually and prevents long edges from being moved as well as fluctuations. In addition, we introduce a constraint to avoid the erroneous visualization issue such as edge-through-terrain. Our main contribution is a 3D edge-bundling algorithm based on the expectation model for origin-destination (OD) data, which provide the end users an interactive means to analyze massive, complex data (Fig. 1).

The rest of this paper is organized as follows. Section 2 reviews work related to edge bundling. In Section 3, we introduce the construction of the expectation model and transform the traditional mechanics for it. Section 4 discusses the edge-bundling and enhancement processes. Then, we present the results on two databases, in 2D and 3D, in Section 5, and design a user study to get feedback from relevant research scholars. In the last section, we summarize and discuss the direction of our future work.

2 Related work

2.1 Edge bundling in 2D

The mechanism of bundling edges in a graph can be dramatically different. The force-directed edge-bundling algorithms simulate physical force. Zhou et al. [18] used Delaunay triangulation to sample the edges of the graph into segments to obtain control points. This non-uniform sampling strategy preserves the topological structure of the graph and allows the midpoint of each sampled segment to be bundled as nodes in the control map. Holten and van Wijk [7] propose an alternative force-directed edge-bundling algorithm, in which the edge is modeled as flexible springs and electrostatic forces exist between the edge pairs. The edge compatibility measures are used to prevent excessive binding. Deformation of edges is calculated iteratively and smoothing is applied.

A different edge-bundling idea implicitly combines edges by routing them through nearby, static control points. Geometry-Based Edge Bundling (GBEB) [3] and edge routing with ordered bundles [14] generate the corresponding mesh geometry by analyzing the structure of the graph and create static control points according to the mesh geometry. An advantage of

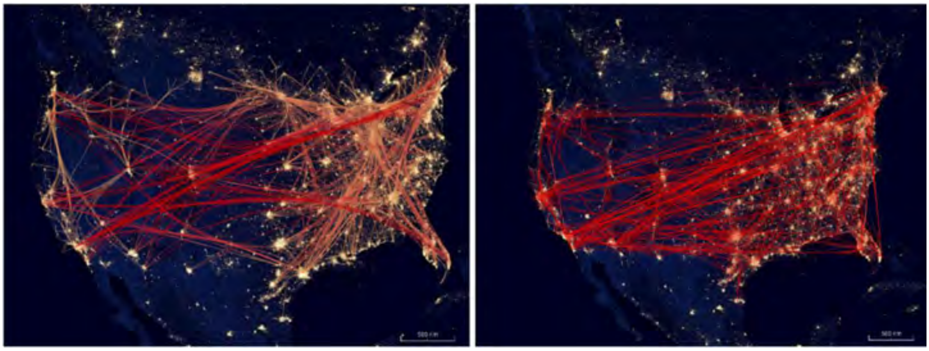


Fig. 1 Results of 3D edge bundling based on the expectation model. **a** Night view of U.S. airlines; **b** Night view of U.S. migration

such methods is less edge-to-edge comparisons. However, generating an appropriate mesh that fits the graph is difficult, which is even more challenge in the 3D scenarios.

The visualization of a data set is closely related to the viewing perspective. When the viewing perspective changes, the 2D edge bundling result has to be regenerated, which is time-consuming and computationally expensive. To reduce the computation time, GPUs are leveraged in recent studies [4, 8, 15], which uses parallel processing and hence greatly improves efficiency. However, extending such methods to 3D scenarios is non-trivial and could result in ambiguity in visualization. It is hence necessary to devise 3D edge bundling methods.

2.2 Edge bundling in 3D

Bottger et al. present the 3D mean-shift edge bundling (3DMSEB) method [2] for functional connectivity graphs in a native 3D brain space, which combines the concept of compatibility from FDEB with the numerical stability and usability of KDE. This method calculates the compatibility between edges and guarantees that only compatible edges are bundled. The method divides edges iteratively and move subdivision points to high-density areas. However, without a constraint of the terrain surface, artificial occlusions occur in the 3D result when geographic data is visualized.

Zielasko et al. introduced the 3D force-directed edge bundling (3DFDEB) [19]. This method is a 3D cluster-based edge-bundling algorithm that extends the FDEB algorithm. The proposed method enhances the stability of the force system and improves computational efficiency. It maintains the edge model and supports the rendering of graphics in different structural styles.

When geographic data are combined with terrain data, the rationality when bundling edges in 3D needs to be considered. Lambert et al. [9] proposed a 3D edge bundling for geographical data visualization (3DEBGD) and applied this method to visualize geographic data. The method extends their previous winding roads (WR) algorithm [10], which transforms the quadtree and Voronoi diagrams used to generate the grid in WR to the corresponding octree and 3D Voronoi diagrams. The algorithm uses the OD data of the route by adding two sets of virtual points to avoid edges passing through the terrain. Although the requirements are met, the result is more like the operation on a sphere, and the addition of virtual points is limited.

3 Expectation model

Considering that the final reflection of bundling is the change in the position of the control points, we design a guidance model for control points that modularizes the guidance factor so that its influence on the movement of the control points tends to be stable.

3.1 The expectation-maximization algorithm

The expectation-maximization algorithm (EM) [12], which is widely used in the machine learning and data clustering processes of computer vision methods, was proposed by T.K. Moon in 1996. EM is a method of finding the maximum likelihood estimate of parameters in the probabilistic model. We apply the core idea of EM to FDEB so that the control points are fully moved at each iteration, thus eliminating the step-setting problem.

3.2 Definition of the expectation model

Definition: The expectation W consists of the expectation shift (desired amount of movement) $W.\boldsymbol{shift}$ and the expectation value (the degree of desire) $W.value$. The expectation shift is what the control point wants to reach the end position under the influence of the guidance factor, and the expectation value indicates the desired intensity. When there are many different guidance factors affecting the same control point, the one with the largest expectation value has the strongest influence.

Multiple expectations can be merged into one, and the combined operation is referred to as the *Mix*. When there are n expectations, i.e., W_1, W_2, $\cdots$, W_n, on a certain control point, they can be fused into

$$W_{mix} = Mix(W_1, W_2, \cdots, W_n). \quad W_{mix}.\boldsymbol{shift} = \frac{\sum_{i=1}^{n}(W_i.\boldsymbol{shift}*W_i.value)}{\sum_{i=1}^{n}W_i.value} \quad W_{mix}.value = \frac{\sum_{i=1}^{n}W_i.value}{n}$$

Note that the *Mix* has the following three properties:

- Multiple expectations generated by the same type of guidance factor can be *Mix* directly.
- Multiple expectations generated by different types of guidance factors need to adjust their expectation values before *Mix*.
- The adjustment of the expectation values needs to be performed by measuring the impact of relevant guidance factors and user demands.

A control point needs to *Mix* all of its expectations into one, and then it should be moved to the final position following the lead of expectation.

3.3 The expectation model of traditional mechanics

According to the definition of the expectation model, we correct the definitions of these two expectations and transform the spring and electrostatic force into homologous expectation models.

As shown in Fig. 2, in terms of the expectation model of the spring force, we set W_{a_0} to represent the expectation generated by a_0 on a_1. When other forces are not considered, the spring force will eventually bring a_1 to the position of a_0. Thus, we obtain $W_{a_0}.\boldsymbol{shift} = \boldsymbol{a_0}-\boldsymbol{a_1}$ and set the expectation. In the force-directed model, the magnitude of the spring force is

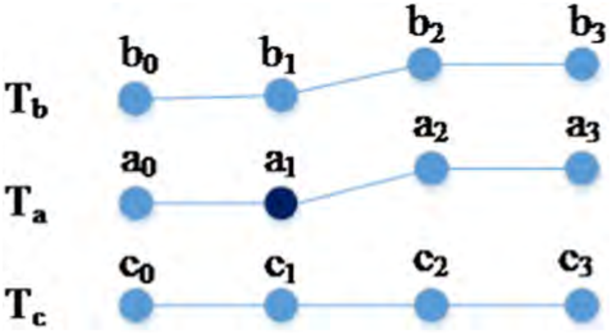


Fig. 2 Sketch map of the force system. T_a, T_b, and T_c are three mutually associative edges. The control point a_1 is affected by the spring force of a_0 and a_2, and is subjected to the electrostatic forces of $b_0 \sim b_3$ and $c_0 \sim c_3$

proportional to the distance between the two control points, which is suitable for the definition of the expectation value. Therefore, we define $W_{a_0}.value = k_{T_a} \times |\boldsymbol{a}_0 - \boldsymbol{a}_1|$, where $|\boldsymbol{a}_0 - \boldsymbol{a}_1|$ refers to the Euclidean distance between a_1 and a_0, and k_{T_a} represents the spring constant of edge T_a. We can obtain the expectation W_{a_2} generated by a_2 on a_1 in the same way. Considering that longer edges can be stretched much farther during bundling, the spring constant should be inversely related to the length of the edge. Thus, we define $k_{T_a} = 1/|T_a|$, where $|T_a|$ is the original length of T_a.

For the expectation model of electrostatic force, we also set W_{b_1} to represent the expectation engendered by b_1 on a_1. Beyond other forces, the electrostatic force will take a_1 to b_1. Similarly, we obtain $W_{b_1}.\boldsymbol{shift} = \boldsymbol{b_1 - a_1}$. In the traditional force-directed model, the magnitude of the electrostatic force is inversely proportional to the square of the distance between the control points. We define $W_{b_1}.value = G(T_b, T_a) \times |\boldsymbol{b}_1 - \boldsymbol{a}_1|^{-2}$. Note that $G(T_b, T_a)$ represents the compatibility between T_b and T_a. Regarding the calculation of the compatibility, the method used in content-importance-based edge bundling (CIBEB) [11] for graph visualization is suitable and will be briefly described as follows.

Define $T = \{T_i\}$ as the set of all edges and $X = \{X_i\}$ as the set of all points. $T_i = \langle X_{O_i}, X_{D_i} \rangle$ represents an edge with vertices X_{O_i} and X_{D_i}. If a non-negative integer p is given, the solution of compatibility is divided into the following four steps:

1) Find p points (not including X_{O_i}) that are closest to X_{O_i} and form a set of compatible points $P_X(X_{O_i}, p)$. Then, calculate the maximum distance from the point in $P_X(X_{O_i}, p)$ to X_{O_i}, denoted as the compatible radius $\sigma_{X_{O_i}}$ of X_{O_i}, and calculate $P_X(X_{D_i}, p)$ and $\sigma_{X_{D_i}}$ in the same way.
2) Calculate the edge set of T_i according to $P_X(X_{O_i}, p)$ and $P_X(X_{D_i}, p)$, which is a set of all edges for its two endpoints in two-point sets of T_i:

$$P_X(X_{T_i}, p) = \left\{ \begin{array}{c} T_m \in T | X_m, X_n \in \{X_{O_m}, X_{D_m}\}, X_m \neq X_n, \\ X_m \in P_X(X_{O_i}, p), X_n \in P_X(X_{D_i}, p), \\ X_m \notin P_X(X_{D_i}, p), X_n \notin P_X(X_{O_i}, p) \end{array} \right\}$$

3) Use the Gaussian kernel function to calculate the compatibility of the endpoint. For a given endpoint X_{O_i} and its compatible point set $P_X(X_{O_i}, p)$, compatibility is calculated as follows:

$$G(X_m, X_{O_i}) = \frac{1}{\sqrt{2\pi}} e^{-\frac{d\left(X_m, X_{O_i}\right)^2}{2\sigma_{X_{O_i}}{}^2}}$$

with $X_m \in P_X(X_{O_i}, p)$, the compatible radius $\sigma_{X_{O_i}} > 0$, and the Euclidean distance $d(X_m, X_{O_i})$ is between X_m and X_{O_i}.

4) For the compatible edge T_m of T_i, the compatibility between T_m and T_i is calculated as follows:

$$G(T_m, T_i) = \left\{ \begin{array}{c} G(X_{D_m}, X_{D_i}) G(X_{O_m}, X_{O_i}), X_{D_m} \in P_X(X_{D_i}, p) \\ G(X_{O_m}, X_{D_i}) G(X_{D_m}, X_{O_i}), X_{D_m} \in P_X(X_{O_i}, p) \end{array} \right\}$$

Note that X_{D_m} may be the compatible endpoint of X_{D_i} or X_{O_i} and that the compatibility calculation should be differentiated. In addition, the compatible radius of the endpoint at each

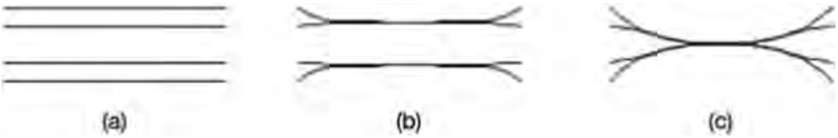


Fig. 3 Diagram of the electrostatic force problem. **a** Original node-link; **b** Incorrect bundling; **c** Expected result

edge may be different; thus, two edges that are mutually compatible may also have different compatibility.

In terms of the electrostatic force, when the control points on the two edges are close, the force becomes infinite. In the force-directed model, when the mutual force between two control points is almost infinite, other forces will be ignored, and the two compatible edges will no longer be affected by other edges; thus, we will obtain erroneous results, as shown in Fig. 3.

If the distance between the two control points is enough small, they should keep this distance during the subsequent iterative process and respond to the forces produced by other points at the same time. The electrostatic force between those two control points only shows the role of keeping them apart, which has no effect on the total environment. To fix this problem, we set the expectation to 0. Therefore, we make the following amendments to W_{b_1}:

$$W_{b_1}.value = \begin{cases} 0, |\boldsymbol{b}_1 - \boldsymbol{a}_1| < minDist \\ G(T_a, T_b) \times |\boldsymbol{b}_1 - \boldsymbol{a}_1|^{-2}, otherwise \end{cases}$$

Our setting for *minDist* depends on the length of the longest edge; assuming it is $n \times 10^x$ with $n \in (0, 10)$, $x \in N$, and then $minDist = 10^{x/3}$.

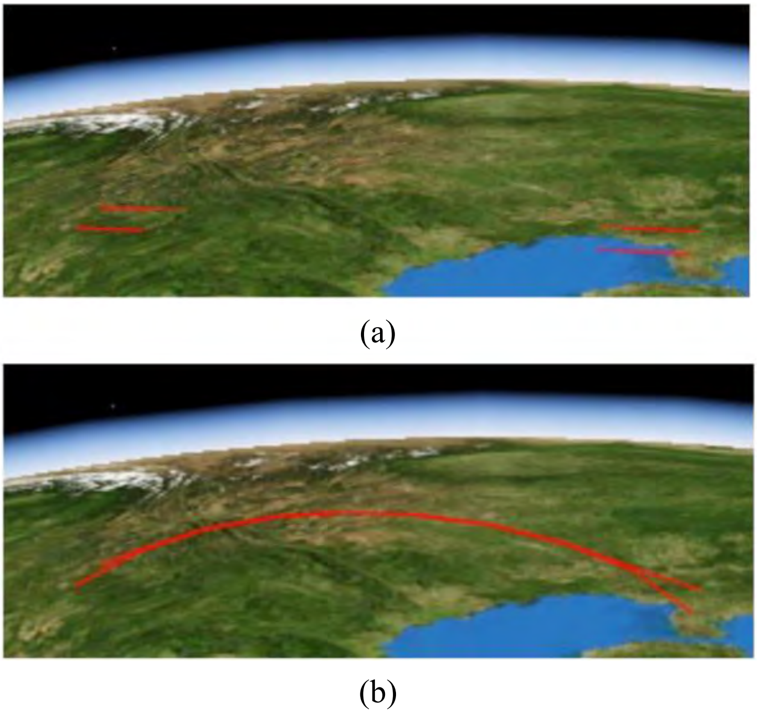


Fig. 4 The effects of centrifugal expectation and spring expectation. **a** Untreated pair of edges; **b** After treatment with two expectations

3.4 Custom expectation model

In the process of route data processing, if different airlines can be used to influence bundling, we can obtain an "Airline force distribution map", and if the type of aircraft is relevant, we will also obtain an "Aircraft trend chart", etc. The user only needs to build a new expectation model that can show the corresponding high-level structure of the diagram.

When geographic information data are visualized in 3D, it is necessary to eliminate the visual clutter to enhance the high-level structure. In particular, if the terrain is incorporated, the edges must avoid being occluded by the terrain or the surface. To meet this requirement, we consider customizing a centrifugal expectation W_f that keeps the control point away from the center of the earth.

Considering that longer edges should float higher, and the control point is farther away from the target height should have a greater centrifugal expectation, our algorithm defines the maximum height H and the centrifugal expectation W_f for the control point a on edge T(R is the radius of the Earth):

$$H = 0.1\left(2R - \sqrt{4R^2 - 4|T|R}\right) \quad = \boldsymbol{a}/|\boldsymbol{a}| \times (H+R) - \boldsymbol{a} \qquad W_f.value = R + H - |\boldsymbol{a}|$$

$$W_f.\boldsymbol{shift}$$

Combining the centrifugal expectation and the spring expectation, a better result is shown in Fig. 4.

3.5 Expectation value adjustments and *Mix*

The expectation value adjustment is normalized. Since the magnitude of different forces varies significantly, such adjustment is needed to prepare for the next *Mix*. After testing, we get a satisfactory result if W_s. *value* = W_s. *value* × 10^2 and W_e. *value* = W_e. *value* × 10^{12} while keeping W_f unchanged.

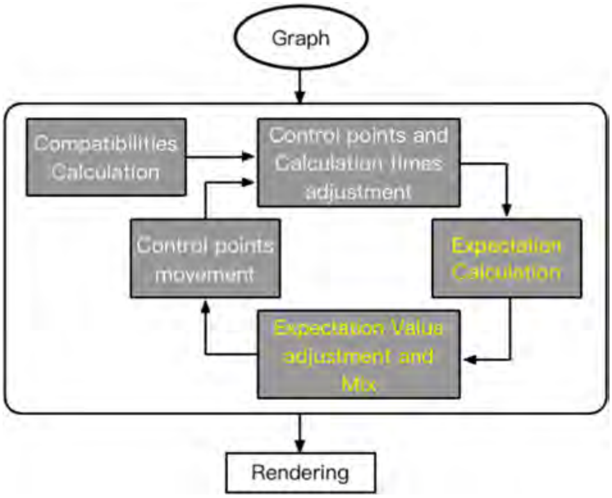

Graph
Compatibilities Calculation
Control points and Calculation times adjustment
Control points movement
Expectation Calculation
Expectation Value adjustment and Mix
Rendering

Fig. 5 Edge-bundling process. The compatibility of each point is calculated and then progressively moved and rendered

Depending on the definition of the expectation, we make adjustments via experimentation. It is difficult to *Mix* multiple expectations at once; thus, we achieve the goal by using a recursive approach: *Mix* two different types of expectations first, then *Mix* the result with one of the remaining expectations, and repeat until complete.

4 Edge bundling based on the expectation model

4.1 Edge-bundling process

A control point may be affected by many expectations simultaneously, and those expectations are constantly changing during the calculation. Thus, the final position cannot be certain at the beginning and needs to be calculated iteratively. The edge-bundling process is shown in Fig. 5.

Decide empirically: The initial values of the parameters are $X = 5$, $N_0 = 1$, and $I_0 = 10$. After each iteration, the number of control points N is doubled, and the number of calculations of each iteration I is reduced by 1/3.

The complexity of our algorithm is the same as that of FDEB in each iteration. However, in the compatibility calculations, we get an overall complexity of $O(nm \log m)$, where m is the number of endpoints; the complexity is $O(n^2)$ for FDEB. When there is a large amount of data or many edges have a common endpoint, i.e., $m \ll n$, our algorithm performs better.

Algorithm 1: Edge Bundling based on Expectation

```
Init $N, I$;
for $n = 0$ to $X$ do
    for $i = 0$ to $I$ do
        for each control point $t$ do
            calculate $W_e$, $W_s$, $W_f$;
            adjust $W_e.value$, $W_s.value$, $W_f.value$;
            $W_t = Mix(W_e, W_s)$;
            $W_t = Mix(W_t, W_f)$;
            $t.\boldsymbol{position} += W_t.\boldsymbol{shift}$;
        end
        Change $N, I$;
    end
end
```

4.2 Smooth and enhancement

We perform Gaussian smoothing for curved edges, which enhances the visual effects. Transparency is used for the edge so that the clusters will have different depths of color. We also set a parameter *minDist* through a global calculation to determine the number of control points k within the distance *minDist* of each edge, and we set the edge width according to k. Let $maxk$ be the largest of all k values; then, the width tw of an edge is:

$$tw = \sqrt{\frac{k}{maxk}} \times 2 + 1$$

5 Results

All experiments are performed on a computer with an Intel Core i7-3770 3.9 GHz quad-core processor running Windows 10 with 8 GB RAM and a GeForce GTX 660 graphics card. For full code please visit our GitHub (URL: https://github.com/SpaceMiao/Expectation-Based-3D-Edge-Bundling).

5.1 3D results

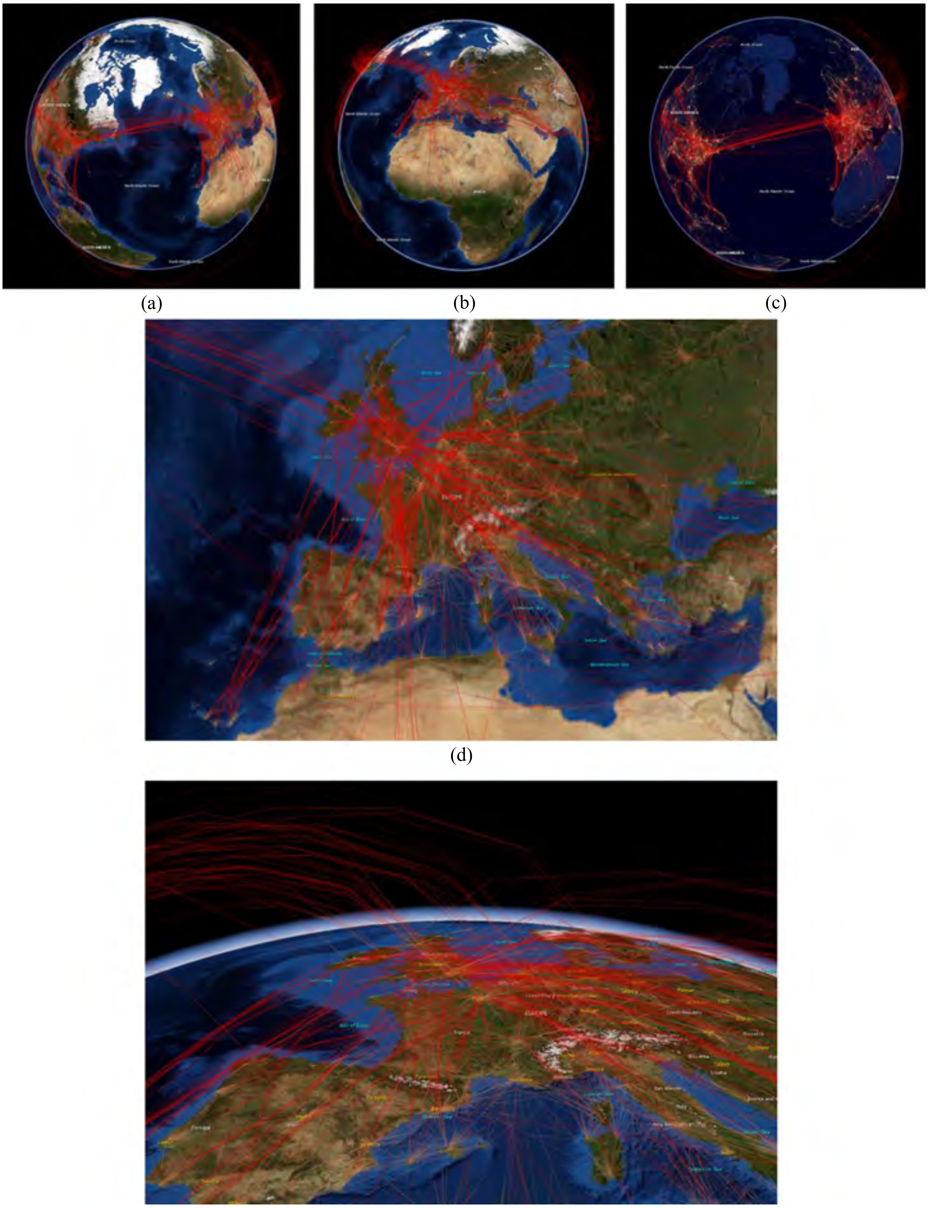
In the experiment, we adopt the OpenFlights Database [16] to evaluate accuracy. The dataset contains 59,036 routes between 3209 airports on 531 airlines spanning the globe. Each entry contains the 3-letter (IATA) codes of the source and the destination airports. We extract the OD data of the route and bundle it, and we combined this dataset with NASA's open-source 3D GIS platform WorldWind [13] to demonstrate the results.

We use the latitude and longitude data to draw all points on the sphere. Then, the first iteration determines the latitude, longitude, and height of all control points of the edge and draw lines between control points. Through continuous iteration, the number of control points on the edge is increased, and the points are moved to the new position and re-rendered. The result is shown in Fig. 6.

The results indicate that our algorithm effectively reduces the visual clutter of the graph, and it is realistic to cluster and bundle important edges. By adding the centrifugal expectation, the edges become suspended on the surface, which correctly shows the results of edge bundling. Since the different lengths of the edges have different heights and colors, the advanced structure of the graph is better displayed. Finally, we show the results from different views. The oblique view (as shown in Fig. 6e) indicates that the edges have a clear tendency to gather after bundling. In night view (as shown in Fig. 6c), we combine the bundling result with the geographic information resources (including geographical texture, city name and other information) loaded online by the WorldWind platform to render and obtained a strong result, from which we can determine the relationship between the distribution and tendency of aviation and cities around the world.

5.2 2D results

To evaluate our proposed algorithm, we also conducted experiments to have 2D visualization using the U.S. airlines and migration datasets, which have been used in the



(a)

(b)

(c)

(d)

(e)

Fig. 6 The results of the OpenFlights Database. **a** and **b** are global overviews from two different perspectives, **c** is the night view, **d** is the specific detail of the European part, and **e** is the oblique view of the European part

evaluation of visualization methods [4, 7, 8, 11, 15, 18]. The U.S. airlines dataset contains 275 vertices and 1925 edges, and the migration dataset contains 1702 vertices and 9780 edges. Using only the spring expectation and the electrostatic expectation, the two methods are evaluated with the same number of iterations, and the results are shown in Fig. 7.

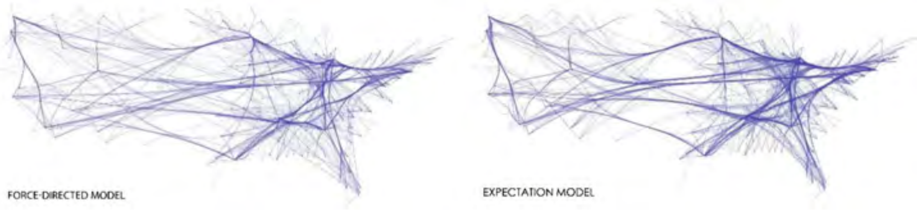


Fig. 7 Comparison of the expectation model and the force-directed model in 2D. The left figure uses FDEB, and the right figure is the result of our algorithm

Both methods suppress visual clutter, and the sub-clusters are mostly the same. Therefore, the expectation model is also applicable to FDEB. Additionally, the expectation model requires only approximately 30% of the calculations of FDEB to achieve the same result.

5.3 Evaluation

Visualization not only presents information but also provides a means of intuitive analysis for users. The user survey evaluations are designed to prove the effectiveness and superiority of our algorithm. We collected and analyzed the user’s judgment and ratings.

A. **A Visual Comparison with 3DEBGD**

To obtain feedback, we used the U.S. airlines data (as shown in Figs. 8, 9 and 10) and migration data (as shown in Figs. 11, 12 and 13) to compare with the rendering under 3DEBGD.

B. **User Study**

We invited 30 participants (5 female, 25 male), 21 to 27 years old, all with normal or corrected-to-normal vision and without any color vision impairment. The participants included

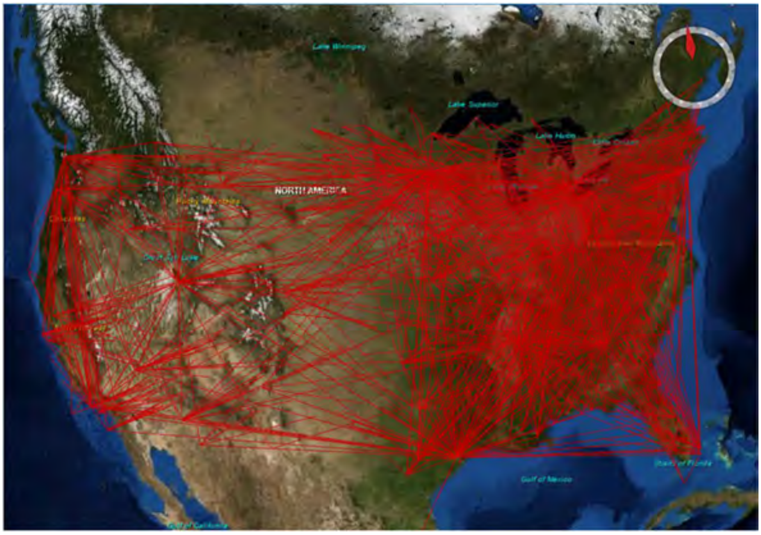


Fig. 8 The effect of airlines raw data with the addition of height

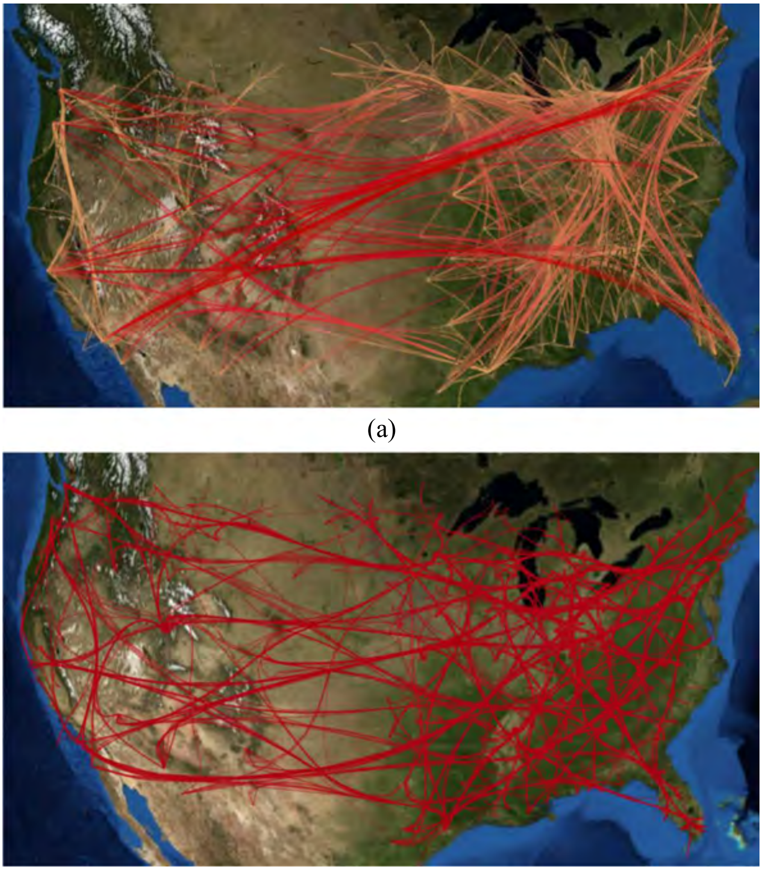
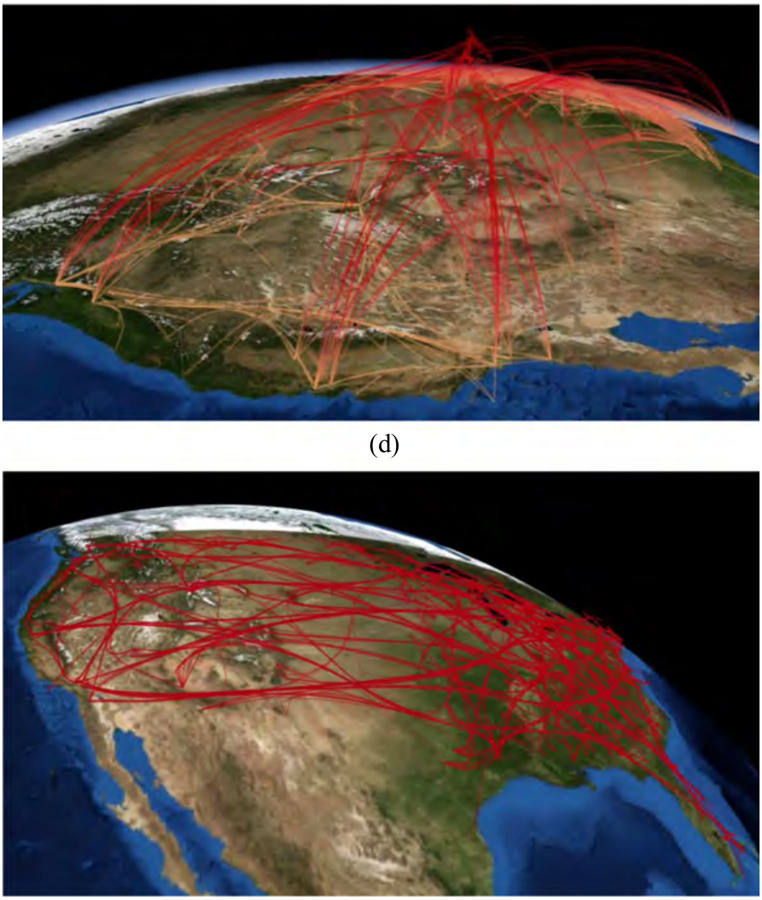


Fig. 9 Top view of two methods in the airlines data, where **a** is our method and **b** is 3DEBGD

undergraduates and postgraduates; 7 participants understood graph visualization, 10 participants knew the concept of visualization, and the rest did not have visual background knowledge. Twenty-two percent of them have participated in the design and development of GIS systems, and 42% have used GIS systems.

The entire survey process included information presentation and a questionnaire. At the start of the survey, we briefly introduced our research project and explained the datasets used in the visualization. Each participant interacted with the visualization of the datasets in a 27 in. LED screen using a 3D GIS platform. This platform allows users to roam the visualization with a mouse. Participants allowed to interact with the system freely at their own pace until they became ready to respond to the questions. We limited the duration of the interaction of a participant to 10 min. Then each participant completed the assessment of the problem on the questionnaire.

We used the 5-level Likert scale for evaluation. Users were asked to rate the questions on the questionnaire using 1–5 points to indicate their satisfaction, where 5 is excellent, 4 is good, 3 is fair, 2 is poor, and 1 is very poor. The questions of the survey are as follows:



(d)

(e)

Fig. 10 Oblique view of two methods in the airlines data, where **a** is our method and **b** is 3DEBGD

Q1: How clear and aesthetic is the view?
Q2: How easy is it to access the information from the view?
Q3: How easy is it to identify the longest connection from the view?
Q4: How easy is it for you to compare connections of airports using the view?
Q5: How easy is it to identify airports using the view?
Q6: How helpful is the view?

After all, participants completed the questionnaire, we calculated the scores and average the scores of the three views separately.

C. Results

Figure 14 shows the average ratings of 30 participants based on the five questions. The average rating of the visualizations using our proposed method is 4.4, whereas the average rating using the 3DEBGD is 3.6. The responses to the question 2 imply that both methods significantly reduce visual clutters. The most significant differences between the 3DEBGD

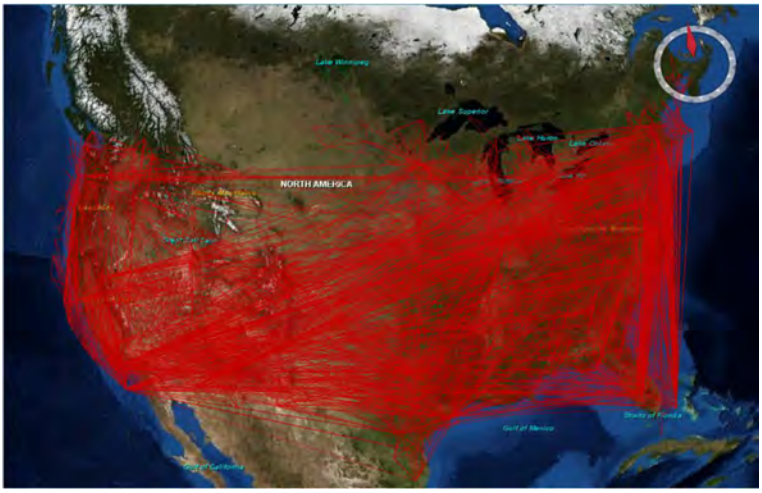
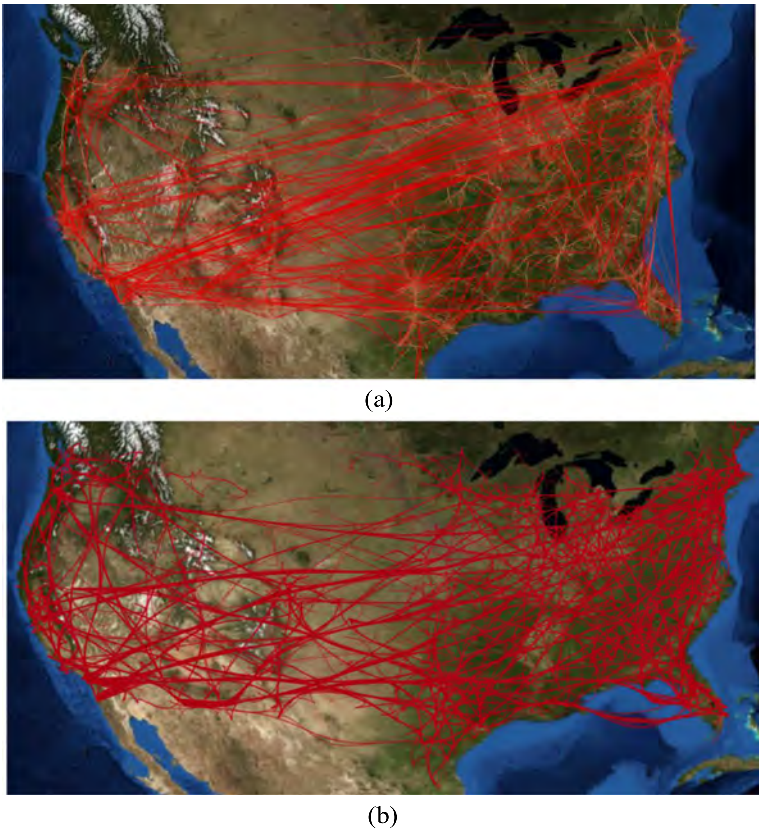
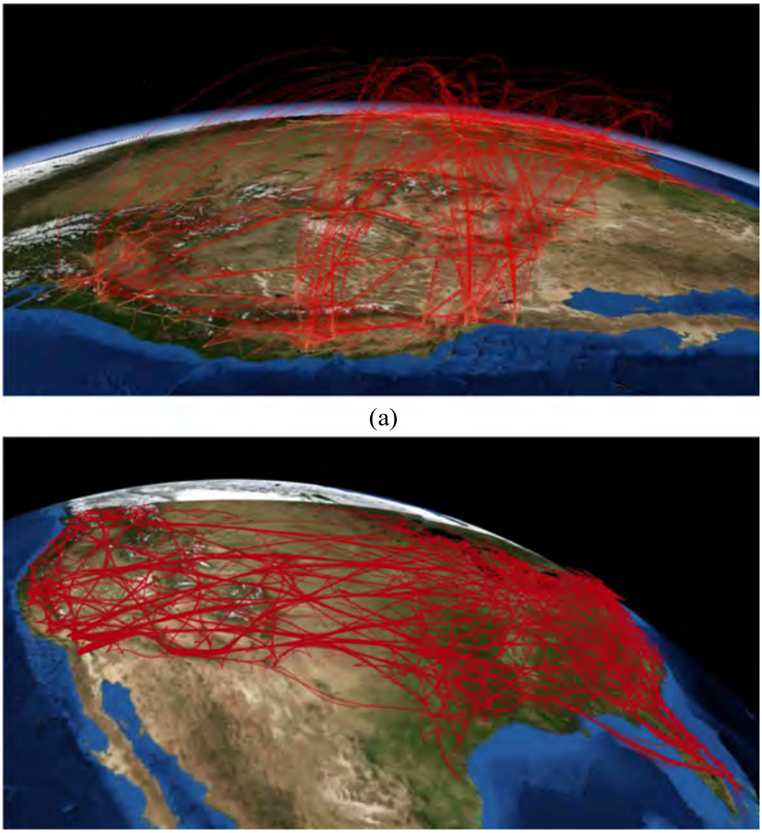


Fig. 11 The effect of migration raw data with the addition of height


(a)

(b)

Fig. 12 Top view of two methods in the migration data, where **a** is our method and **b** is 3DEBGD



(a)

(b)

Fig. 13 Oblique view of two methods in the migration data, where **a** is our method and **b** is 3DEBGD

method and the proposed one lie in the responses to questions 3, 4 and 5, which focus on the derived information of the visualization. The airline dataset, there are a large number of short-distance flights. The color used in our visualization is proportional to the elevation, which makes it easy to identify the origin and destination of frequent regional flights. In addition, the oblique view of the 3D visualization allows users to perceive the airport with

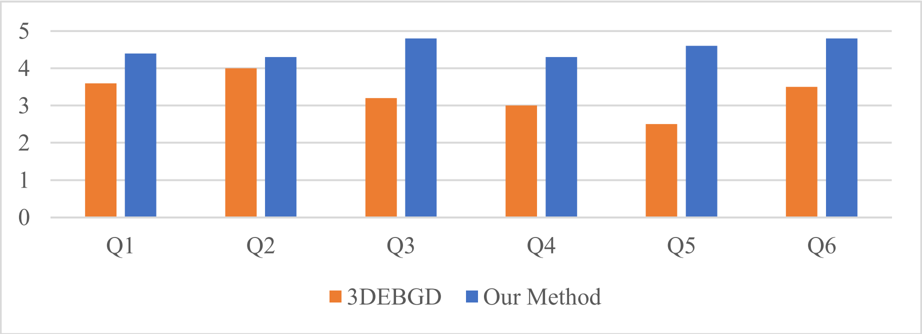


Fig. 14 The questionnaire results of our method and 3DEBGD

ease. In contrast, the added virtual points in the visualization of the 3DEBGD method are rendered on the surface along with the endpoints, which clutters the visualization and makes it difficult to distinguish flight routes and airport. The twisted routes to long connections from multiple bundles make it difficult to see prominent trends. Positive responses to question 6 of our method confirms that the 3D arching routes make the visualization more realistic and close to human preference.

6 Conclusions

In this paper, we propose an edge-bundling algorithm based on the expectation model that eliminates the use of the step parameter in the iterative calculation, transforms the spring force and electrostatic force into corresponding expectation models and successfully completes 3D edge bundling with the combination of 3D GIS datasets by adding the centrifugal expectation. Using flight data, we evaluate the algorithm in 3D. The experimental results show that our 3D edge-bundling algorithm is suitable for general node-link diagrams and produces better results than another method. This algorithm enhances the display of high-level edge patterns while reducing visual clutter. Most importantly, it allows for customization through the addition of rules to reflect different priorities.

In our future work, we plan to improve from the following three aspects:

1) Optimize the bundling process of the algorithm to speed up the processing and find the best choice of parameters and their transformations.
2) Add more interactions so that relevant researchers can get the information easily.
3) Add more information and design a more aesthetic visualization without compromising the authenticity of the data.

Acknowledgments This work was supported in part by the Natural Science Foundation of Anhui Province of China under Grant 1708085MF158, in part by the National Natural Science Foundation of China under Grant 61472115, in part by the Visiting Scholar Researcher Program at North Texas University through the State Scholarship Fund of the China Scholarship Council under Grant 201706695044, and in part by the Key Project of Transformation and Industrialization of Scientific and Technological Achievements of Intelligent Manufacturing Technology Research Institute of Hefei University of Technology under Grant IMICZ2017010.

References

1. Bondi AB (2000) Characteristics of scalability and their impact on performance. In: Proceedings of the 2nd international workshop on software and performance. ACM, pp 195–203
2. Böttger J, Schäfer A, Lohmann G, Villringer A, Margulies DS (2014) Three-dimensional mean-shift edge bundling for the visualization of functional connectivity in the brain. IEEE Trans Vis Comput Graph 20(3): 471–480
3. Cui W, Zhou H, Qu H, Wong PC, Li X (2008) Geometry-based edge clustering for graph visualization. IEEE Trans Vis Comput Graph 14(6):1277–1284
4. Ersoy O, Hurter C, Paulovich F, Cantareiro G, Telea A (2011) Skeleton-based edge bundling for graph visualization. IEEE Trans Vis Comput Graph 17(12):2364–2373
5. Herman I, Melan¸con G, Marshall MS (2000) Graph visualization and navigation in information visualization: a survey. IEEE Trans Vis Comput Graph 6(1):24–43

6. Holten D (2006) Hierarchical edge bundles: visualization of adjacency relations in hierarchical data. IEEE Trans Vis Comput Graph 12(5):741–748
7. Holten D, Van Wijk JJ (2009) Force-directed edge bundling for graph visualization. In: Computer graphics forum, vol 28. Wiley Online Library, pp 983–990
8. Hurter C, Ersoy O, Telea A (2012) Graph bundling by kernel density estimation. In: Computer graphics forum, vol 31. Wiley Online Library, pp 865–874
9. Lambert A, Bourqui R, Auber D (2010) 3d edge bundling for geographical data visualization. In: Information visualisation (IV), 2010 14th international conference. IEEE, pp 329–335
10. Lambert A, Bourqui R, Auber D (2010) Winding roads: routing edges into bundles. In: Computer graphics forum, vol 29. Wiley Online Library, pp 853–862
11. Lu Q, Kunle M (2016) Content importance based edge bundling for graph visualization. CAD&CG, 28(11): 1899–1905
12. Moon TK (1996) The expectation-maximization algorithm. IEEE Signal Process Mag 13(6):47–60
13. NASA: World wind (2006). URL https://worldwind.arc.nasa.gov/?root=java. Accessed Oct 2017
14. Pupyrev S, Nachmanson L, Bereg S, Holroyd AE (2011) Edge routing with ordered bundles. In: International symposium on graph drawing. Springer, pp 136–147
15. Telea A, Ersoy O (2010) Image-based edge bundles: simplified visualization of large graphs. In: Computer graphics forum, vol 29. Wiley Online Library, pp 843–852
16. The OpenFlights Database (2009). URL https://openflights.org/. Accessed Oct 2017
17. Ware C, Mitchell P (2008) Visualizing graphs in three dimensions. ACM Trans Appl Percept 5(1):2
18. Zhou H, Yuan X, Cui W, Qu H, Chen B (2008) Energy-based hierarchical edge clustering of graphs. In: 2008 IEEE pacific visualization symposium. IEEE, pp 55–61
19. Zielasko D, Weyers B, Hentschel B, Kuhlen TW (2016) Interactive 3d force-directed edge bundling. In: Computer graphics forum, vol 35. Wiley Online Library, pp 51–60

Guibing Yang is currently pursuing a master's degree in computer science and information from Hefei University of Technology, China. His current interests include moving object trajectory and visualization.

Kunle Ma received the master degree in computer science and information from Hefei University of Technology, China. His current research interests include visualization, cooperative computing, and image processing.

Xiaohui Yuan received a B.S. degree in Electrical Engineering from the Hefei University of Technology, Hefei, China in 1996 and a Ph.D. degree in Computer Science from the Tulane University, New Orleans, USA in 2004. He is currently an Associate Professor at the Department of Computer Science and Engineering at the University of North Texas and a Visiting Professor at the China University of Geosciences, Wuhan, China. His research interests include computer vision, data mining, machine learning, and artificial intelligence. Dr. Yuan is a recipient of Ralph E. Powe Junior Faculty Enhancement award in 2008 and the Air Force Summer Faculty Fellowship in 2011, 2012, and 2013. He is a senior member of IEEE.

Jie Li received the master degree in computer science and information from Hefei University of Technology. He is an associate professor in the School of Computer and Information at Hefei University of Technology. His primary research interests include Computer control technology, management information system, urban water supply monitoring and dispatching and intelligent water service.

Qiang Lu received the PhD degree in computer science and information from Hefei University of Technology. He is an assistant professor in the School of Computer and Information at the Hefei University of Technology, and member of Key Laboratory of Industrial Safety and Emergency Technology of Anhui Province. His primary research interests include visualization, cooperative computing, and image processing. He is a member of the IEEE and CCF.