Global Illumination for Fun and Profit

Roy G. Biv*
Starbucks Research

Ed Grimley†
Grimley Widgets, Inc.

Martha Stewart‡
Martha Stewart Enterprises
Microsoft Research

ABSTRACT

Group in a box (GIB) is a graph-drawing method designed for visualizing the group structure of graphs. GIB allows the group sizes and both the inter- and intra-group structures to be viewed simultaneously. Several GIB variants have been proposed to realize each design goal. However, their advantages and disadvantages from the perspective of human cognition have not been studied. Herein, we conducted a user evaluation, including eye-tracking analysis, of four GIB variants to clarify them. From the evaluation, we found some trade-offs among each method for the type of user task.Z The eye-tracking analysis provided some clues as to why these results were generated.

Index Terms: Human-centered computing—Visualization—Visualization techniques—Treemaps; Human-centered computing—Visualization—Visualization design and evaluation methods

1 INTRODUCTION

In the area of graph drawing, numerous methods have been proposed, each aiming at a different visualization goal. Some of the methods are effctive for the analysis of real-world graph data, such as social networks and web graphs. They are often characterized as being so-called complex networks [**?**], a feature of which is their community structure [**?**, **?**]. A network with a community structure contains certain groups that comprise several nodes and have dense within-group edges and sparse between-group edges. Additionally, real-world data often provide information about the groups to which the nodes belong. For example, consider a Twitter network data, it has users as nodes and edges connecting couples of users if one replies to or mentions the other. Users can be divided to some groups in such a network by its feature. When a network become larger and with higher density, it can be more challenging to for an observer to understand this network data because of the high number of nodes, edges, and groups resulting in visual clutter. To analyze the real-world data, an effective visualization technique for understanding the group structure of the network is essential.

There are various methods to group related nodes in a network based on a feature of that such as the topology, some attributes the nodes have, or the combination of themw [**?**]. A topological method generate groups so that the relationships among nodes in a single group are stronger than those beween nodes in different groups. The latter method take some common attributes to the nodes. For example in the case of Twitter network, there are user's geographical location, religion, and interests. General methods to illustrate these groups visually discriminate them by colour or shape of nodes, which is not effective way to make it easy to understand when a network is large.

Group in a box (GIB) is a graph-drawing method designed for a network in which a node belong to a group, and visualize the group structures in graphs. GIB arranges all nodes in a group into a box with an area proportional to the number of nodes in the group so that each group are visualized separatedly. It then lays groups inside their boxes with a force-directed, circle, grid or other available layout. Using a GIB layout enables the simultaneous visualization of group structures, intra-group relationships, inter-group relationships and the sizes of the groups in a graph. It is especially expected that placing groups in indivisual boxes helps understand intra-group relationships. There are several variants of GIB layout [**?**, **?**, **?**], and each of them aims different purpose originally. Although a few computational experiments [**?**, **?**] have evaluated these GIB variants, they have not been evaluated through user experiments to the best of our knowledge. Some network measures used in the computational experiments(e.g., the number of edge crossings) are known to affect the readability of a graph drawing [**?**, **?**, **?**, **?**], but other elements related to readability in GIB visualizations make user experiments important.

Herein, we aim to ascertain which GIB variant is the most effective. We report on the evaluation of four GIB variants, namely ST-GIB, CD-GIB, FD-GIB, and TR-GIB, with four types of user tasks. The tasks are designed based on the studies by Vehlow et al. [**?**] and Saket et al. [**?**] to view each layout's features from various perspectives. We measure the task accuracy and completion time for each task. We also collect the eye-tracking data of participants during the tasks.

Eye-tracking is a tequnique for recoding gaze data and often used user experiments in the field of visualization and human-computer interaction science. It is useful for analyzing tasks in this case because the gaze data of participantas can give us some insights into why the layouts affect the results [**?**]. Along with the accuracy and completion time of tasks, we also analyze the eye-tracking data to reveal why one layout is better than the other layout. Specificlly, we focus on what element in a layout makes it easy or difficult to solve tasks.

One of our main contribution is to evaluate 4 GIB layouts and find the best GIB. For the another thing, we enable to avail these 4 GIB layout for public. Our website (url???) provide GIB image, as well as further analysis of the network. It can help reseachers analyze their network data and possibly give them new findings.

2 RELATED WORK

2.1 Graph Drawing mehtod for group structure

The purpose of graph drowing method is to design a layout diagram for complex network. Thomas et al. proposed a method using force-directed placement, in which consider attractive and repulsive force to produce an optimal equilibrium [**?**]. This force-directed method has been widly used and applied to various forms. There are several other approaches [**?**, **?**, **?**, **?**]. Although these general visualizations taske network structure in to account, they do not visualize the complexity of group structure explicitly, so are often challenging for the understanding of a clustered graph.

On the other hand, the recent popularity of the network visulalization has developed some techniques designed for the network with group structure. In large scale network data, the importance of community detection is known and several approach, often based on force-directed layout, have been invented. One traditional method to illustrate the group structure of a graph is varying the colour [**?**] or the shape of node, but often not effective in the real world network.

*e-mail: roy.g.biv@aol.com
†e-mail: ed.grimley@aol.com
‡e-mail: martha.stewart@marthastewart.com

Vehlow et al. provides an survey paper about visualization of group structure in graphs [?]. They discribes a taxonomy of such visualization and also that of tasks used in user-study experiments for them. The visualization taxonomy is clussified by two ways, vertex visualization taxonomy and vertex group structure. The latter depends on whether a network can be characterized by either or both of group overlapping and hierarchy of the structure or not. A node in a network with group overlapping can belong to multiple groups at the same time. A hierarchical network has some hierarchical structure among groups. They introduced some visualizations designed for a network with group structure at each category in the taxonomy.

The GIB, Group-In-a-Box, layout is also a visualization for group structure categorized as with no group overlapping presented in [?, ?, ?]. There are 4 major variants of GIB which we tackle in this research, ST-GIB, CD-GIB, FD-GIB, FD-GIB. This layout have an advantage of visualizing each groups clearly separeted, which allow t,o observe a relationship between two node in the same group easily. It can alsp provide the information of group size quickly as a form of the rectangular's area. This categorized as a method for the network without group overlapping in Vehlow's taxonomy, and ST-GIB and TR-GIB can be applied for a network with hierarchcal structure. Therefore an example of network data in which GIB is effective is a Twitter data with some groups defined as the users location, because the location can not make an overlap. This time we focus on these GIB layouts and aims to asceritain which layout is the best in our user study.

2.2 Evaluation of Network Diagram

3 GIB Layouts

In this section, we describe the four evaluated GIB variants, namely ST-GIB, CD-GIB, FD-GIB, and TR-GIB, the examples of which are shown in Fig. 1.

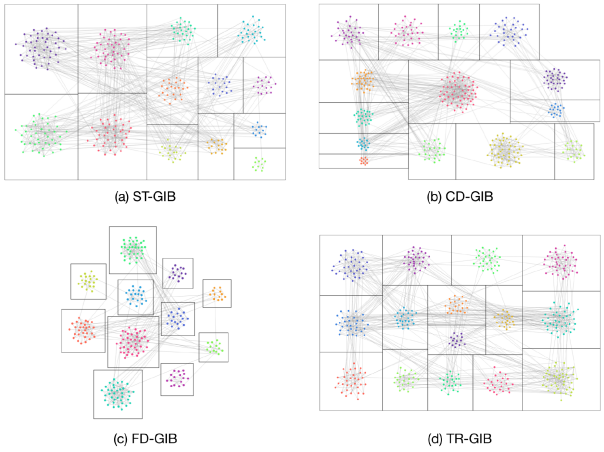

Figure 1: Examples of group-in-a-box (GIB) layouts.

3.1 ST-GIB

Squarified treemap GIB (ST-GIB) (Fig. 1(a)) is a layout based on squarified treemaps by Bruls et al. [?] and proposed by Rodrigues et al. [?]. Bruls' method was originally for treemapping, which is a visualization method that takes rectangular regions and numerical columns as inputs to divide a region into tiles with areas proportional to their values [?].

In ST-GIB, the groups are taken as vertices in treemaps and shown in the shape of tiles that have nodes belonging to its group in it. The treemap algorithm facilitates a space-filling arrangement with low-aspect-ratio boxes, which is important for analyzing the rectangle content [?]. However, the ST-GIB layout uses only the squarified treemap algorithm to arrange group networks in rectangles, so it does not consider the relationships among nodes when the tiles are placed. This problem causes link overlaps, which considerably hamper the understanding of networks [?, ?, ?, ?]. To arrange tiles using this method, we utilized the Python library squarify (https://github.com/laserson/squarify), which employs Bruls' algorithm.

3.2 CD-GIB

Chaturvedi et al. [?] presented a croissant-and-doughnut GIB (CD-GIB) (Fig. 1(b)). They developed this layout to improve ST-GIB to be able to consider network information connecting a node to another of a different group. Concretely, they worked on arranging tiles based on G-degree and G-skewness. A group's G-degree is defined as the number of other groups it is connected to, and G-skewness refers to the fraction of nodes present in the two most-connected groups (groups with the highest G-degree). One layout is then chosen from croissant-GIB, doughnut-GIB, or ST-GIB according to the total number G of groups and the G-skewness. Chaturvedi et al. defined the criteria for using the three methods properly, which are also used by us.

The most connected box with the highest G-degree is placed close to the center to allow these methods to account for links among groups, so we expect better readability than that achieved with ST-GIB because of fewer edge crossings [?, ?, ?, ?]. However, the aspect ratio of a box tends to be worse, which can affect the rectangle content's capability of being analyzed [?].

3.3 FD-GIB

Chaturvedi et al. [?] also provided the FD-GIB layout (Fig. 1(c)). This approach explicitly shows the inter-group relationships. The group boxes are arranged using a force-directed layout run on the whole network, where the vertices represent entire groups and the edges between them show the links between groups. They then draw the group boxes on this initial layout centered at the group node's position. This layout can cause group overlaps, which are removed using the PRISM method [?]; we use this method to eliminate all overlaps.

Chaturvedi et al. used the Harel–Koren fast multiscale layout [?] to arrange the rectangles in FD-GIB, but we use the D3.js force simulation (https://d3js.org/) because it is easy to obtain and produce good results. However, as reported by Chaturvedi et al. [?], according to the experimental evaluations of several layout algorithms performed by Hachul et al. [?, ?], there are several good choices, such as the high-dimensional embedding approach [?] or the algebraic multi-grid method [?].

This layout has the benefit of showing the aggregate topology clearly, but it wastes more screen space. The area of each group must be smaller than those of the others, is is expected more difficult to see the relationships in a single group. However, this layout can maintain a unit box aspect ratio, so we expect participants to easily recognize the size difference between boxes.

3.4 TR-GIB

In previous work, we developed TR-GIB to minimize the weighed sum of distances between groups by reordering the sibling nodes in ST-GIB [?]. An example of this is shown in Fig. 1(d). This method looks similar to ST-GIB, but the tiles are reordered. Treemaps, e.g., ST-GIB, have a tree-like structure, so vertices with the same depth in it can be reordered. This layout aims to reduce the sum of the link distances, which iresults in less edge crossings that cause a misunderstanding. The coordinates of this layout can be obtained by reordering those of the ST-GIB layout to minimize the proximity. Specifically, the sum of the distances between two groups is

weighted according to the number of edges between the two groups and this sum is taken as the group proximity.

This layout has the minimized group proximity, so the edge length is shortened, which leads to edge crossings lesser than those with ST-GIB. We expect this layout to have the advantages of ST-GIB (i.e., good aspect ratio and screen efficiency) and the advantage of less edge overlaps.

4 User Experiments

In this study, we conducted user experiments with an eye-tracking system for various tasks to reveal which layout is superior. The GIB layouts that we compared are ST-GIB, CD-GIB, FD-GIB, and TR-GIB. In addition to the accuracy and completion time of tasks, we measured the eye-tracking data to evaluate the layouts quantitatively. Using the eye-tracking system allows us to discuss the results in detail [**?**].

4.1 Tasks and Stimuli

We began the user study by setting up our tasks according to Vehlow et al. [**?**] and Saket et al. [**?**]. Vehlow et al. introduced four task taxonomies regarding the evaluation of clustered graph visualization, namely group-only tasks (GOTs), group vertex tasks (GVTs), group edge tasks (GETs), and group network tasks (GNTs). Among these tasks, we can use GOTs, GVTs, and GNTs for our GIB evaluation; GETs are for networks whose edges are grouped. Saket et al. also provided four types of tasks, namely GOTs, group node tasks, group link tasks, and GNTs, all of which can be applied to ours, but we note that the GNTs of Vehlow et al. include the group node tasks and GNTs of Saket et al. Vehlow et al. and Saket et al. also described examples of each task type. As a group node task, there is the task of finding the group with the most nodes. We selected our tasks from these examples and exposed the participants to the following four tasks.

- Task 1 (GOT): How many groups are present in this graph?
- Task 2 (group-node task) : Which group has the maximum (minimum) number of nodes?
- Task 3 (group-intra-link task) : Which group has the maximum (minimum) number of links?
- Task 4 (group-inter-link task) : Which group has the maximum number of links connecting its nodes outside the group?

We chose two GNTs because we aimed to assess the layout's capability of representing both group intra-links and group inter-links. A group intra-link connects a node to another node in the same group, whereas a group inter-link connects a node to another node of another group. Because GIB is a method for visualizing the relationships among groups and in a group, the GIB layout method should be good at showing both the intra- and inter-links.

In tasks 2 and 3, because both small and large boxes should be shown comprehensively, we set two tasks each, namely finding the maximum and minimum number of nodes. We changed the question, namely maximum or minimum, at the half of total trials of the tasks for tasks 2 and 3.

4.2 Hypotheses

We constructed some hypotheses regarding our tasks. From our computational experiment, we know that there are differences among these four GIB variants, so we expect these layouts' variation to affect our tasks.

- Hypothesis 1: Participants can complete task 1 best with FD-GIB. Although FD-GIB is not space efficient, there are margins between the boxes, and therefore, the groups are separated more clearly in FD-GIB. Participants can recognize the number of groups earlier.
- Hypothesis 2: In task 2, FD-GIB achieves the best result. FD-GIB has good aspect ratio (1.0); therefore, a side of a box corresponds to the square root of its area. Participants can easily discriminate among box sizes.
- Hypothesis 3: In task 2, CD-GIB makes it difficult to find the biggest (smallest) box. The box aspect ratio is poor with this layout, making it less suitable for the task.
- Hypothesis 4: In task 3, TR-GIB makes it easiest to find the answer. This layout performs well in terms of aspect ratio and screen space efficiency, so each network in a box can occupy more area than it can with CD-GIB or FD-GIB. In addition, because we expect few edge crossings, we can see the intra-edges with fewer interruptions caused by inter-edges.
- Hypothesis 5: In task 3, FD-GIB makes it difficult to solve the task because its boxes are smaller those those in other layouts. If a box is small, the links in it are short, making it challenging to determine the number of links visually.
- Hypothesis 6: In task 4, TR-GIB and FD-GIB are superior to the other layouts. TR-GIB is good at edge overlaps, making it easy to recognize the inter-links. FD-GIB wastes space but provides areas for inter-links only, not boxes. We expect it to be easy to observe inter-links with two layouts.

4.3 Data and Layout Generation

We used random data for these experiments, generated by a procedure similar to that used by Onoue et al. [**?**]. Each graph consists of multiple groups with some strong edge connections between certain group pairs. There are two differences between this method and that proposed by Onoue et al. The first is the way of setting the number of groups randomly from a normal distribution with m_{mean} and m_{stdev} ranging from m_{min} and m_{max}. The second is that we determine the number of vertices in a vertex set V_i, which we determine based on the random numbers from a normal distribution with v_{mean}, v_{stdev}, and a minimum of v_{min}. We calibrated the graph generation parameters listed in Table 1 to bring our data closer to the actual Twitter data reported by Chaturvedi et al. [**?**]. We set these parameters through several processes. First, we calibrated them to make the numbers of groups, nodes, and links correspond to those reported by Chaturvedi et al. However, this yielded more nodes and links than we could interpret, so we reduced them by multiplying v_{mean} and v_{stdev} by 0.4 and p_{in}, p_{bridge}, and p_{out} by 0.3. In this way, we obtained the graphs shown in Fig. 1.

m_{mean}	m_{stdev}	m_{min}	m_{max}	V_{mean}	V_{stdev}	V_{min}	p_{in}	p_{group}
11.4	5.4	6	11	21.0	14.12	4	0.0858	0.06

Table 1: Parameters used for data generation

After generating the data, we applied the four layouts to them separately, thereby obtaining the tile arrangements. The four targeted layouts arrange only boxes, so we had to determine the node coordinates in a box. Although there are many ways to place the nodes in a box, we used the force layout. This method is widely used in graph drawing wherein the nodes are subjected to repulsive between-node forces, attractive forces between adjacent nodes and their center of gravity, and gravity from the center of the group tile to which they belong.

We also used straight lines for edges. Other than straight lines, there are options such as edge bundling. However, we aimed to calculate the total number of edge crossings, which affects the readability of a graph drawing, as mentioned in Section 3.3, so we selected straight lines.

4.4 Experiment Design

There were 30 questions for each task and each layout (in total, 120 questions per task and 480 trials for the experiments). Participants might have become confused if we changed the task for every trial, so the participants performed the same task continuously until it ended. Additionally, the participants might have become accustomed to the data if we showed them network diagrams comprising the same data, so we did not show all data repeatedly. We generated 120 data for each task based on Section 4.3, and we had 480 data in total. We randomized the question order for each task, so the network layout shown to participants changed at each trial. We also randomized the task order to mitigate the effects of learning and fatigue. Participants took a brief break of roughly 30 s after every 20 questions and a relatively longer break at one task, 120 questions. The longer break was up to 5 min, after which we re-checked the eye-tracking calibration. We empirically set the color as gray and width of edges, as shown in Fig. 1.

4.5 Study Procedure

We used a repeated-measures study design, and the variables of interest were as follows.

- **Layout of network diagram:** We used four GIB variants, namely ST-GIB, CD-GIB, FD-GIB, and TR-GIB.
- **Task:** We performed four tasks as described in Section 5.1.

The participants were asked to reveal their age, gender, and field of study in advance. Next, they read a manual about biometric measuring, eye-tracking, and each GIB layout. We then explained these experiments in detail, followed by a tutorial to check whether they could interpret the networks and solve our tasks. The trial data of the tutorial differed from those in the actual experiments but nevertheless provided a practical guide to the experimental procedures.

Once the participants completed all the above steps, they proceeded to the actual experiments. They worked on the four tasks in random order to compensate for the leaning and fatigue effects, and 120 continuous trials existed in the same task. For tasks 2 and 3, we asked the participants two questions that maximum or minimum: the first 60 trials asked for the maximum, whereas the last 60 trials asked for the minimum, where we permuted the trials so that each half has 15 networks for each layout respectively in these tasks.

There was no time limit on the tasks, and the answers were sent to our server. The participants were instructed to focus on answering correctly. If they focused on answering quickly, we were likely to encounter higher error rates and more chaotic gaze trajectories, which was not the intention of the present study. The participants chose the solution by click and confirmed it by pressing the keyboard.

We conducted these eye-tracking experiments in a room with artificial illumination and few objects. The participants' mobile phones were switched off to reduce distractions during the experiments. The eye movements were recorded using a Tobii Pro X3-120 eye-tracking system with an ASUS screen resolution of $2,018 \times 1,920$ pixels. The participants sat in front of a display at a distance of 65 cm. For the analysis software of the eye tracker, we utilized I-VT filter to determine whether a gaze was a fixation or a saccade, as described in [**?**].

4.6 Participants

We used a within-subjects study design with 20 participants, all of whom were Japanese (eight women and 12 men). Their average age was 20.8 years, with the youngest participant being 18 and the oldest participant being 24. All were students at our university: four were studying engineering, whereas the other participants were mostly students with various majors. All participants had normal or corrected-to-normal color vision. Each participant was compensated with the sum of 3,000 yen. The experiment duration was 1.5–2 h, including the explanation and breaks.

5 Experimental Results

5.1 Generated Layouts

There are various known metrics that can affect the readability of graph drawing. We computed three measures to obtain insights into the user study, namely the number of edge crossings, the ratio of screen space wasted, and the mean group aspect ratio. We chose these three metrics with reference to those used by Chaturvedi et al. [**?**], although we used edge crossings rather than their edge box overlap. We describe these metrics in detail below.

- **Edge Crossings**

  This metric can reduce the readability of graph drawing and should therefore be small. In particular, a long inter-edge can overlap with many other inter-edges and intra-edges of boxes on the edge's way. We applied the same force-directed layout to all boxes to measure the number of interrupting factors for our data, as generated in Section 3.3.

- **Amount of Screen Space Wasted**

  A two-dimensional screen space-filling layout has the potential to be comprehensible [**?**]. In GIB, more efficient screen space causes boxes to become bigger and each network to be drawn with more screen space. When screen space is limited, links can be too short to interpret and networks can become visually unappealing. For effective visualization, screen space should not be wasted.

- **Mean Group Box Aspect Ratio**

  Bruls et al. emphasize that rectangles that are more square-like have advantages such as screen efficiency, easier comparison of box size, and accuracy of presentation [**?**]. They also note that thin rectangles cause aliasing errors, whereas square items are easier to detect and point at. Because we use force-directed layouts that depend on the gravity from the center of the box, the network nodes tend to be arranged spherically, thereby yielding a good aspect ratio close to unity.

Layout	Edge crossings	Screen space wasted	Mean aspect ratio
ST-GIB	12,536 (10,496)	0.0 (0.0)	1.40% (0.10)
CD-GIB	13,383(11,424)	0.123 (0.073)	2.01% (0.40)
FD-GIB	12,808 (11,140)	0.697 (0.051)	1.0% (0.0)
TR-GIB	11,381 (9,797)	0.0 (0.0)	1.40% (0.10)

Table 2: Computation results: mean (standard deviation)

Table 2 summarizes the calculation results. In graph drawing, we demand fewer edge overlaps, lower aspect ratio, and less wastage of screen space. The results show that TR-GIB had the fewest edge crossings, thereby indicating that TR-GIB is the most effective layout for reducing edge overlaps.

Regarding the ratio of wastage of screen space, it was low for ST-GIB and TR-GIB, followed by CD-GIB, as summarized in Table 2. Furthermore, FD-GIB yielded the best aspect-ratio, followed by ST-GIB and TR-GIB. ST-GIB and TR-GIB performed well in terms of space efficiency and aspect ratio, whereas CD-GIB and FD-GIB performed well in terms of either space efficiency or aspect ratio.

5.2 Task Results

In this subsection, we describe the results of the user experiments, excluding the eye-tracking data, which are discussed in the following subsection. The results of the user study are summarized in Table 3. We conducted one-way ANOVA with four layouts against each task, for which p was 0.05 against the accuracy and the task completion time, as summarized in Table 4. We used one-way ANOVA based on the reasoning that each layout has its advantages and disadvantages and that all tasks are independent.

Layout	Task 1	Task 2	Task 3	
ST-GIB	98.1% (3.987 s)	89.6% (2.535 s)	67.4% (3.606 s)	6
CD-GIB	98.2% (4.527 s)	76.9% (2.761 s)	67.2% (3.843 s)	5
FD-GIB	96.7% (4.919 s)	82.9% (2.427 s)	78.8% (3.382 s)	5
TR-GIB	98.1% (4.493 s)	83.6% (2.713 s)	72.8% (3.845 s)	63.5% (5.128 s)

Table 3: Results of user study: mean accuracy (mean completion time)

For task 1, there is a significant difference in task completion time: ST-GIB performed the best and FD-GIB performed the worst in this task. Participants counted the number of groups fastest with ST-GIB, and this is against hypothesis 1. With FD-GIB, they took the longest time to recognize the answer.

There are significant differences in both accuracy and completion time for task 2. The participants had the highest accuracy with ST-GIB and the lowest accuracy with CD-GIB. Regarding task completion time, FD-GIB performed the best. Hypothesis 2 is refuted, and hypothesis 3 is confirmed.

FD-GIB resulted in the highest scores for both accuracy and time in task 3; however, we observed significant differences among them, thereby refuting hypothesis 5. Hypothesis 4 is confirmed because TR-GIB resulted in the second highest accuracy.

For task 4, there are no significant differences in terms of accuracy and time, so hypothesis 6 is not confirmed.

	Task 1	Task 2	Task 3	Task 4
Accuracy	0.2377	2.579e−7	1.303e−5	0.3039
Time	9.009e−16	7.571e−5	4.706e−5	0.06275

Table 4: P-values of ANOVA for accuracy and completion time

5.3 Eye-tracking Results

We collected the eye-tracking data during the user study and analyzed it based on [**?**] wherein Andrienko et al. described the visual analytics methodology with respect to tasks and gave guidelines for method selection. We selected trajectory maps, as shown in Fig. 5.3, which are useful for understanding the overall spatial pattern of movements but are sometimes ineffective with excessive amount of data. However, our eye-tracking data for graphs are not enormous as there are only 20 participants, so we selected trajectory maps to visualize the eye movements of the participants. We also computed the total gaze count, as summarized in Table 5.3. The gaze count is strongly related to the task completion time listed in Table 3. According to Pearson's correlation analysis, there is a strong relationship ($r = 0.993, p = 1.65\text{e}{-14}$).

6 Discussion

Hereinafter, we discuss the above results using the eye-tracking data. Using the eye-tracking system, we aimed to clarify why one layout is better or not than the other layouts.

In task 1, participants answered the slowest with FD-GIB, for which we observed that the gaze trajectories covered the boxes in

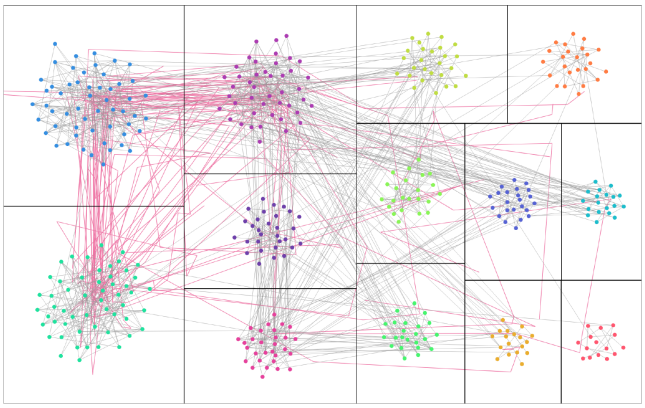

Figure 2: Example of a trajectory map. Pink lines represent gaze trajectories.

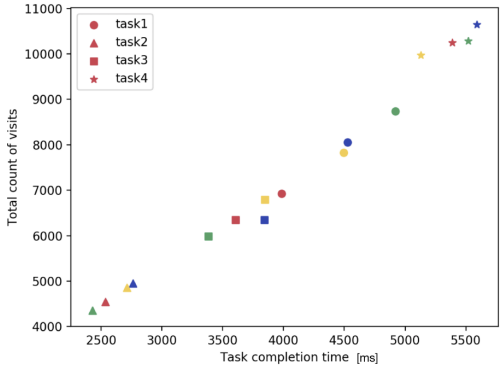

Figure 3: Relationship between task completion time and total gaze count: ST-GIB (red), CD-GIB (blue), FD-GIB (green), TR-GIB (yellow).

all layouts even though the space occupied by boxes was small with FD-GIB. The total gaze count suggests that the reason for this difference is multiple visits. With FD-GIB, participants considered multiple visits to count the number of groups, which indicates that FD-GIB takes a considerable amount of time to count the number of groups, thereby supporting the result. We assume that participant re-counting was the cause of these many visits, which means that FD-GIB makes counting difficult, mainly because of the wasted space.

In task 2, the accuracy was highest with ST-GIB and lowest with CD-GIB. The layout with the best aspect ratio, namely FD-GIB, was not as accurate as ST-GIB or TR-GIB, so we conclude that ST-GIB and TR-GIB result in aspect ratios that are sufficiently good to understand their box sizes. We suppose that ST-GIB yielded the highest accuracy in this task because with our algorithm, the layout arranges the boxes in the order of their sizes

CD-GIB and TR-GIB resulted in many gaze visits, and this may have caused the slowness of task completion in task 1. However, we cannot consider this as the reason for the accuracy difference because TR-GIB yielded relatively high accuracy. On observing the trajectory maps, we speculate that the reason was the more dispersed gaze with CD-GIB. With CD-GIB, in addition to the poor aspect ratio, boxes with similar areas can be arranged far apart, whereas ST-GIB places big boxes close to each other as expected, making it easy to compare the box sizes. The poor aspect ratio and dispersed boxes might have confused participants, thereby increasing the total number of visits.

FD-GIB yielded the best accuracy and completion time in task 3, with TR-GIB yielding the second highest accuracy. This result corresponds to that of computation at 4.1: FD-GIB and TR-GIB result in relatively few edge crossings, which interfere with readability [**?**, **?**, **?**, **?**].

On analyzing the trajectory map, we found that the main strategy in task 3 was to compare some large or small boxes. A bigger/smaller box tends to have more/fewer links, so participants extracted some big or small boxes along the task for the first step, followed by the next step of comparison of the links in the boxes. Here, FD-GIB seems to have outperformed TR-GIB because the accuracy for task 2, involving an extraction process similar to that in task 3, was almost the same. We suppose that the reason why FD-GIB made it easier than TR-GIB despite having more edge crossings was the box size. Our task required only a box with the most or the fewest links, so participants could determine the answer by only seeing the density of links as the depth of color. In our graphs, the parts with many links look dark and those with less links look white because of the link color (i.e., gray). We suppose that it is easier to discern the light and dark parts with FD-GIB, which results in smaller boxes. When we set another task to see intra-links, such as finding the vertex with the highest degree, TR-GIB can be effective because its bigger boxes can show each relationship clearly.

In task 4, there are no significant differences although the number of edge overlaps differs. The gaze trajectories were observed along the inter-edges (shown in the Appendix). We believe that undesirable long inter-edges make it easier to find the answer in this task, contrary to our hypothesis 7, which produces edge crossings. The long edges stand out visually; thus, boxes with many long edges can be detected easily.

Overall, FD-GIB and TR-GIB made good achievements in our tasks, whereas CD-GIB did not perform well in all tasks. ST-GIB performed well in tasks 1–3 even though it does not consider the relationships between groups. Task 3 showed the difference between FD-GIB and TR-GIB, which have boxes of varied sizes, so we recommend choosing either of these two layouts properly for the purpose of users.

7 Conclusions and Future Work

In this study, we evaluated four GIB layouts by means of a user study with eye tracking. We determined which layout is best and obtained evidence to support the results. The best layouts were FD-GIB, with good aspect ratio and edge crossings, and TR-GIB, which also has good aspect ratio (to some extent) and edge crossings. Furthermore, our user experiments reveal their higher performances. ST-GIB performed well in terms of understanding the size of each box; however, it was poor in interpreting links. We concluded that FD-GIB and TR-GIB are effective, with both layouts having their own advantages and disadvantages. FD-GIB performed well in determining the number of links, a blurred information, whereas TR-GIB was better in representing more concrete relationships, e.g., links between two specific nodes.

Regarding the limitations of this work, other algorithms or methods could be used to generate random data, although we aimed to make them realistic. In particular, we used only one set of parameters in the user study; thus, other cases would be of interest. Additionally, there are several other ways of expressing nodes and edges and arranging nodes, which may influence the results. We found that undesirable long edges were effective in task 4; therefore, another task may lead us to further results.

Our massive eye-tracking data could be used for further analysis in future work. Many methods that were not used in this study are available for dealing with eye-tracking data, which may lead to novel discoveries. Moreover, we are now preparing a public web-based GIB visualization system. We hope that anyone can access GIB variants easily and obtain new findings with these layouts.

References

[1] P. Isenberg, F. Heimerl, S. Koch, T. Isenberg, P. Xu, C. Stolper, M. Sedlmair, J. Chen, T. Möller, and J. Stasko. vispubdata.org: A Metadata Collection about IEEE Visualization (VIS) Publications. *IEEE Transactions on Visualization and Computer Graphics*, 23, 2017. To appear. doi: 10.1109/TVCG.2016.2615308

[2] G. Kindlmann. Semi-automatic generation of transfer functions for direct volume rendering. Master's thesis, Cornell University, USA, 1999.

[3] Kitware, Inc. *The Visualization Toolkit User's Guide*, January 2003.

[4] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, Aug. 1987. doi: 10.1145/37402.37422

[5] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995. doi: 10.1109/2945.468400

[6] G. M. Nielson and B. Hamann. The asymptotic decider: Removing the ambiguity in marching cubes. In *Proc. Visualization*, pp. 83–91. IEEE Computer Society, Los Alamitos, 1991. doi: 10.1109/VISUAL.1991.175782

[7] G. Wyvill, C. McPheeters, and B. Wyvill. Data structure for *soft* objects. *The Visual Computer*, 2(4):227–234, Aug. 1986. doi: 10.1007/BF01900346