GION: Interactively untangling large graphs on wall-sized displays

Michael R. Marner[1], Ross T. Smith[1], Bruce H. Thomas[1], Karsten Klein[2], Peter Eades[2], and Seok-Hee Hong[2]

[1] University of South Australia
Email: {michael.marner, ross.smith, bruce.thomas}@unisa.edu.au
[2] The University of Sydney
Email: {karsten.klein, peter.d.eades, seokhee.hong}@sydney.edu.au

Abstract. Data sets of very large graphs are now commonplace; the scale of these graphs presents considerable difficulties for graph visualization methods. The use of interactive techniques and large screens have been proposed as two possible avenues to address these difficulties.This paper presents GION, a new skeletal animation technique for interacting with large graphs on wall-sized displays. Our technique is based on a physical simulation, and aims to enhance the users' ability to efficiently interact with the graph visualization for exploratory analysis. We conducted a user study to evaluate our technique against standard operations available in most graph layout editors, and the study shows that the new technique produces layouts with less stress, and fewer edge crossings. GION is preferred by users, and requires significantly less mouse movement.

1 Introduction

Graphs provide a versatile model for data from a large variety of application domains, including biology, finance, telecommunication, software engineering, and social sciences. Graph visualization helps scientists and engineers to understand critical issues in these domains. However, the depth of understanding depends on the quality of the drawing. Automatic graph layout methods are developed for computational efficiency and quality, i.e. readability. These methods however can only optimize a few criteria in combination, and it is impossible to define a quality measure that allows to create optimal layouts for all graphs, tasks, and observers. Moreover, the size of relevant data sets for analysis has grown exponentially over the last years. For example, data from social networks, biology, and finace continue to grow at a rate that is not accommodated by current methodologies.

While some layout algorithms are capable of laying out graphs with hundreds of thousands of nodes in a few seconds [4], data sizes from practice are still a challenge in a number of ways:

- There is a trade-off between computational resources and layout quality. For example, algebraic methods run quickly but in many cases give poor results [6], while stress minimization [5] gives good quality layouts but is too slow for interactive work on large graphs.

– Existing methods do not scale well visually. A standard screen with a few megapixels cannot faithfully display graphs of a few million edges.
– An underlying problem lies in the optimisation criteria for layout algorithms for large graphs. For small scale graphs, criteria such as the number of edge crossings have been successfully used and validated [13, 14]. However, all commonly used algorithms for large graph layout ignore these criteria and generally use an optimisation criterion based on a notion of energy or stress [4].
– Readable large graphs might not be sufficient for understanding, as requirements differ based on the application and the task at hand. Moreover, the user can interactively explore different regions of the graph which are not known in advance, e.g. to compare or investigate the local structures, while keeping their global context.

This paper makes the contribution of a new interaction technique, GION, for manipulating layouts of large graphs. GION is novel by employing a physics engine to simplify the process of interacting with large graphs, treating the graph as a set of connected rigid bodies. The physics engine provides smooth animation for the user while interactively laying out the graph, and this animation improves the understanding for the user of the graphs underlying structure. When the user moves a cluster, the connected clusters are also moved, as if they were connected in a chain. Our contribution is validated with a user study conducted to evaluate the effectiveness of the technique in a *graph untangling* task.

The remainder of this paper is structured as follows. Section 2 describes previous research related to this paper. Section 3 describes the details of the new graph interaction technique. Section 4 outlines the user study conducted to evaluate the new interaction technique, with the results presented in Section 5 followed by a discussion of these results in Section 6. Finally, the paper concludes with a discussion of future work.

2 Background

While readability of graph drawings has been a topic of research for decades, and there is a wealth of papers on the evaluation of drawing quality, e.g. [7, 13], the research has focused mainly on task based performance for diagrams of small to medium size. Several well established quality criteria for graph layouts exist. The most prominent one is the number of crossings, which was verified to be an impediment for the human understanding of small graphs in empirical experiments [13]. Further well established quality criteria are angular resolution, edge length deviation, and stress. Recently, Huang et al. [7] suggested that it is often better to make compromises between aesthetics, instead of trying to satisfy one or two of them to the fullest. It has however not yet been investigated if the results obtained for small graphs can be extended to huge graphs or if different quality criteria have to be employed.

Recently, Dwyer et al. [3] compared user-generated and automatic graph layouts, where users were asked to optimize the layout for aesthetics and social network analysis tasks. In their study, users that were asked to optimize a graph layout for an analysis task used the term “untangling” to describe their process. In contrast the term “untangle” here is not task related but used in a relatively informal sense: a user “untangles” a graph drawing when they improve the layout, in the subjective opinion of the user. In

particular, we did not ask the participants of our user study to optimize any pre-specified quality metric. Indeed, our long-term goals include discovery of users' metrics.

Our new technique employs animation techniques found in modern computers games, and many applications apply animation techniques to enrich the look and feel of the user interface. Animations to the interface smooth the rough edges and sudden transitions common in many current graphical interfaces, and strengthen the illusion of direct manipulation that many interfaces strive to present [16]. Animation improves a user's understanding of the direct manipulation of the data by better portraying such concepts as constraints, relationships, and connectivity. These are powerful cues for the direct manipulation of large graph structures.

Ball, North, and Bowman [1] evaluated interaction techiques for large display visualisations. They tracked physical navigation in 3D space via the participants head with a VICON system. Their experiments found with increased size of the display, there was more physical navigation. When combined with the reduced performance time on large displays, they found a compelling suggestion that physical navigation was also more efficient. They also found that physical navigation was preferred over virtual navigation.

Peck, North, and Bowman [12] defined a new 3D interaction technique, *multiscale interaction*, which associates the user's scale of perception to their scale of interaction. Multiscale interactions exploit the user's physical navigation in front of a large display to directly control the scale of interaction, while adjusting their scale of perception. Overall, they found evidence that multiscale interaction is a natural behavior, and this technique can be useful in interaction design for large high-resolution displays.

Skeletal animation (see, for example, [8]) is a well established technique in graphics. An object is modelled as a mesh, with "bones" as links between "joints". Movement of the mesh in between keyframes can be computed using methods of *inverse kinematics* [17]. The skeletal animation technique is mainly used to animate people and animals, but Merrick et al. [9, 10] investigate application to graph interaction. Their work, however, is limited to very small graphs.

3 GION: Graph Interaction Operation for Nodes

To untangle a graph G, the user has to rearrange nodes by dragging them to new positions. Moving nodes one by one is time consuming. With thousands to millions of nodes, the human resources required are too large both for untangling in practice and for evaluation experiments like the one in this paper. Large graphs thus need interaction methods for untangling that move more than one node at a time. The *GION* technique uses ideas and off-the-shelf software from skeletal animation to simplify the process of interacting with large graphs. More specifically, GION adapts a physics engine to move many nodes at a time. GION treats the graph as a skeleton, where bones simulate edges, and joints simulate nodes. However, the simple approach of representing every edge as a bone and each node as a joint does not scale to handle large graphs. Thus large graphs are clustered to enable the user to move large numbers of nodes at once. The physics engine treats clusters as rigid bodies connected by joints. The effect of this approach is that connected clusters move as a chain. This section describes the interaction technique in detail and provides rationale for design choices.

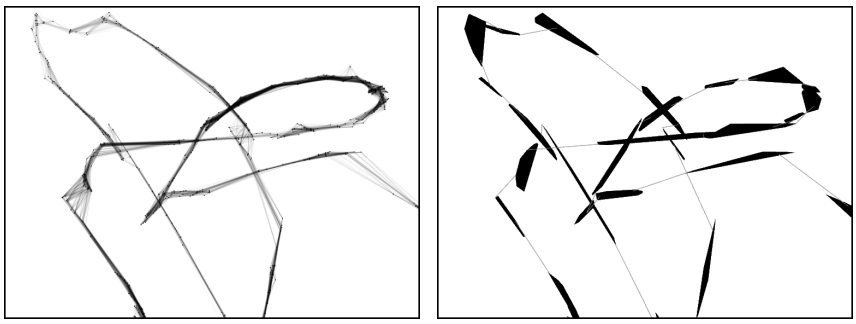
Fig. 1. Untangling a graph layout on the tiled wall display.

3.1 Graph Clustering

Allowing the user to move larger chunks of the graph at a time can help to reduce this effort in case these chunks are specified in a way that supports the user's untangling process. Graph clustering partitions the nodes of the graph into clusters, that is, disjoint node sets, aiming to have high *cohesion* (that is, many intra-cluster edges), and low *coupling* (that is, few inter-cluster edges).

Many different clustering algorithms are available. We use a fast and simple clustering method based on random walks [2]. This algorithm aims to detect dense local substructures using a random walk based graph traversal. Roughly speaking, random walks tend to stay within a highly cohesive substructure with a high probability [2]. This algorithm allows to influence the number of clusters over parameter settings; this property is helpful for tuning the interaction fidelity. As the cluster boundaries created by the random walk approach can be somewhat fuzzy, we apply a Kernigan-Lin style postprocessing technique [18] that flips the cluster affiliation of single nodes that are connected more strongly to a different cluster than to the cluster they are affiliated with.

Clusters as Rigid Bodies: Each cluster in the graph is represented as a polygonal rigid body. The polygon shape is calculated by taking the convex hull of the vertices in the cluster, and then simplifying the polygon down to a maximum of eight vertices. This simplification greatly improves the performance of the simulation at runtime. The polygon is given physical properties that drive the simulation: **1) Damping** reduces the velocity of rigid bodies when in motion. **2) Density** determines the mass of the polygon, and thus its momentum when in motion. **3) Static Friction** prevents rigid bodies from moving unless a minimum threshold force is applied. This is important in limiting the number of nodes that move in response to the movement of a node. **4) Collision** In most physics simulations, rigid bodies can collide. For our purposes, collisions are disabled and bodies can pass through each other. Physics engines such as that provided by Box2D allow specification of control parameters for the above properties; for GION, we chose values for these parameters from experience.



(a) (b)

Fig. 2. (a) A section of a graph, and (b) physics system representation.

Edges, clustered edges, and bones: If there is at least one edge between two clusters C_1 and C_2, then we say that C_1 and C_2 are *linked*. If two clusters are linked, then we place an elastic *bone* between them; physically this acts to approximately maintain a specified distance between the two clusters. The bone has a *length* that defines this distance. In GION, the length of a bone is held constant during a single user operation, but varies from one operation to the next, as explained below. The physics engine provides two parameters to define the elasticity of a bone: *frequency* and *damping ratio*. In GION, the elasticity is held constant throughout.

3.2 User operations.

Users interact with the graph using the mouse to drag clusters into new positions. Each GION user operation consists of three steps: 1) The user selects a node with a button-down mouse event. 2) The user moves the mouse. Mouse movement is applied to the physics simulation as a force that acts on the cluster containing the node selected by the user. Using a force provides intuitive feedback about the connections from the selected cluster; heavily connected clusters are more difficult (require more mouse movement) to move than clusters with only a few connections. 3) The user releases (button-up) the mouse. GION then re-sets the length of each bone to the distance between its endpoints.

GION differs significantly from previous skeletal animation methods in that the length of each bone can vary from one user operation to the next. In a classical skeletal animation (say of a human walking), bones have constant length; but GION introduces some elasticity to the bone length. This provides more information to the user, as they can more easily see how the graph is connected. However, we found through informal pilot studies that users felt as though they did not have enough control over the layout of the graph. Users became frustrated when clusters, connected by constant length bones, would rebound back towards their initial positions.

To improve this behaviour, GION resets all bone lengths when the user releases the mouse. The distance between two clusters at that point in time becomes the new defined distance for the bone: thus the clusters stay in their positions when the user releases the mouse. This improves the control that users have over the layout. It also gives the

users the ability to stretch out parts of the graph layout by quickly moving different clusters to new positions. A future extension to this technique would allow users to explicitly shrink bones, bringing clusters together. This extension was not included in the evaluation.

We also make use of the clustering information to color the graph, based on the degree of physics bodies. The clusters with the highest degree (most linked) are drawn in red, and clusters with the lowest degree are colored green. The remaining clusters are colored as a gradient from red to green based on their cohesion. Coloring the graph in this way allows users to quickly identify which parts of the graph will be easiest to move into more desirable locations.

4 Evaluation

We conducted a user study to evaluate the benefits of the physics-based graph interaction technique. We chose an *untangling* task for the experiment. Participants were shown a series of graph layouts, and were asked to 'untangle' the layout to better show the overall structure. Untangling was chosen as the task because it would require many mouse operations to complete, and resulting graphs could be compared to the initial layout. The experiment is a 2x2, within participant, repeated measures design. The conditions tested were interaction mode: *physics* or *normal*, and coloring: *colored* or *plain*. Participants used the GION technique for the *physics* condition. The *normal* interaction allowed users to move nodes to new locations one cluster at a time, emulating movement operations commonly used in graph layout software.

The hypotheses tested in the experiment are as follows:

H1 *Physics* interaction leads to lower stress than *normal* interaction.
H2 *Physics* interaction leads to fewer edge crossings than *normal* interaction.
H3 *Colored* graphs would have lower stress than the *plain* graphs.
H4 *Colored* graphs would have fewer edge crossings than the *plain* graphs.
H5 *Physics* interaction requires less mouse movement than *normal* interaction.
H6 *Physics* interaction requires fewer clicks than *normal* interaction.
H7 *Physics* interaction leads to lower stress than the starting layout.
H8 *Physics* interaction leads to fewer edge crossings than the starting layout.
H9 *Physics* interaction is preferred by users.

The following data were collected during each trial of the experiment: mouse movement, in millimetres on the videowall, mouse clicks, and snapshots of the graph layout (captured every five seconds). From the graph layout snapshots other properties of the layout could be calculated and analysed. Participants were asked to fill out a questionnaire at the end of the user study session. Participants answered questions 1-4 for both the physics and normal interaction conditions, and questions 5-6 for the color condition. All questions were answered using a visual analogue scale. The participants were asked to rank the interaction conditions in order of preference, and comment on strategies used for untangling the graphs.

1. Moving graph clusters into new positions was {very easy - very hard}

2. Untangling the graphs was {very easy - very hard}
3. The interaction mode made the graphs {more understandable - less understandable}
4. With the results of untangling the graphs, I was {very happy - very unhappy}
5. When deciding which clusters to move first, the coloring made it {easy - hard}
6. Coloring the graph made untangling {fast - slow}

4.1 Graphs

We selected graphs that come from a real world application where graph visualization is used for data analysis and the graph size and layout quality requirements pose a challenge for state-of-the-art layout methods. Our graph set consists of RNA sequence graphs that are used for the analysis of repetitive sequences in sequencing data [11]. They have been created by running pairwise alignment on genomic sequence reads, and to represent reads as nodes and large overlaps between reads as edges. Eight graphs were chosen for the experiment. Participants untangled all graphs, with the conditions randomised for each graph.

We applied the Fruchterman-Reingold algorithm `FR` to obtain the initial layouts used in the experiments. Our goal here was to start with a layout that did not reveal the overall graph structure completely. A completely random layout might pose a too difficult challenge. Using a layout generated by `FR` allows the user to identify starting points for untangling while leaving enough space for improvement based on individual preferences. Graph properties are provided in Table 1. The set of graphs can be downloaded from `http://wcl.ml.unisa.edu.au/graph-untangling/graphs.zip`. All graphs have high local density, and a sparse global structure that allows to create layouts far from hairballs that are showing the structure well.

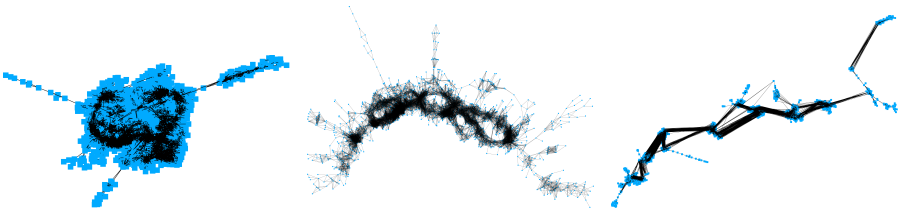
Graph	# nodes	# edges	density	avg deg.	clus. coeff.	avg sh. path	diameter
A	1159	6424	5.5	11.1	0.65	19.5	59
B	1748	13957	8	16	0.64	17.9	63
C	1785	20459	11.5	22.9	0.61	10.7	41
D	3010	41757	13.9	27.7	0.67	26.4	77
E	4924	52502	10.7	21.3	0.65	36	121
F	5452	118404	21.7	43.4	0.73	46.6	216
G	5953	186279	31.3	62.6	0.72	56.2	163
H	7885	427406	54.2	108.4	0.69	24.4	55

Table 1. Overview on the graph set used for the experiment

4.2 Experimental Procedure

Each participant completed the experiment in a single session. Participants were first asked to complete a graph theory quiz. This quiz asked simple graph structure questions and was designed to allow results from participants with different levels of knowledge to be compared, and did not affect the rest of the experiment. Participants were given

instructions on how to interact with the system, and that they would have two minutes to untangle each graph. Specifically, participants were shown an example starting layout and untangled layout (shown in Fig. 3) and told "to untangle the graph. This involves moving parts of the graph to new locations in order to make the underlying structure of the graph clear". Following this, the untangling trials began. The display provided instructions informing the participant of the conditions of the next trial (for example, *physics colored*). The participant clicked the mouse to begin the trial and the display changed to show the initial graph layout. The participant then had two minutes to best untangle the graph. After two minutes, the video wall went blank while the next trial was loaded. After eight trials the participants were asked to complete the subjective questionnaire and the session was concluded.



(a) (b) (c)

Fig. 3. (a) The starting layout for graph B. (b) Layout for graph B using stress minimization. (c) Final layout created by one of the participants.

5 Results

Sixteen participants completed the experiment, recruited from staff and students from the University of South Australia, and the general public. Participant ages ranged from 23 to 57, with a mean age of 33. Five of the participants were female, and all but one of the participants were right handed. The mean score for the graph theory quiz was 74.22% (std. dev 25.19). Pearson Correlation analysis was performed to see how quiz score affected results. A significant correlation was found between quiz score and change in edge crossings for the normal/color condition, $r = -.505[-.876, .453], p < 0.05$. Quiz results did not affect any other conditions. All other quantitative results were analysed using a 2x2 repeated measures ANOVA.

5.1 Mouse Usage

There was a significant main effect on interaction mode, $F(1,15) = 36.586,\ p < 0.001$. There was significantly less mouse travel in the *physics* condition compared to *normal*, with 59294.661mm (SE 5001.180) of mouse travel for the *physics* condition compared to 76227.881mm (SE 4153.657) for the *normal* condition. The graph coloring did not

produce a significant effect on mouse travel, and there was no significant interaction between interaction mode and graph coloring.

There was a significant main effect on the interaction mode, $F(1,15) = 6.279,\ p < 0.05$. Participants made fewer mouse clicks for the *normal* condition than *physics*, with mean mouse clicks of 38.563 (SE 3.235) for the *normal* condition compared to 43.109 (SE 3.036) for the *physics* condition.

There was also a significant main effect on the graph coloring, $F(1,15) = 11.614,\ p < 0.01$. Participants made significantly fewer mouse clicks for the *plain* condition than *colored*, with mean mouse clicks of 39.125 (SE 2.685) for the *plain* condition compared to 42.547 (SE 3.367) for the *colored* condition.

5.2 Graph Layout Analysis

Layouts were analysed for stress and edge crossings. The results presented here are represented as a ratio of change in values compared to the starting values of the initial graph layout, i.e. $Result = (EndValue - StartValue)\,/StartValue$. Using a ratio of change allows comparisons between graph layouts of different sizes, numbers of clusters, and vastly different initial stress and edge crossing values. In all conditions, stress was higher after participants interacted with the graph. Of the 128 trials conducted in the experiment, only two resulted in lower stress values than the initial conditions. However, there was a significant main effect on interaction mode, $F(1,15),\ p < 0.05$. The physics based interaction produced layouts with significantly less stress than the normal interaction mode, with mean stress change of 22.884 (SE 2.772) for the *physics* condition compared to 95.763 (SE 14.955) for the *normal* condition. Graph coloring did not produce a significant effect on graph stress, and no significant interaction between interaction mode and graph coloring was found. Edge crossings were also higher after participants interacted with the graph. Sixteen of the 128 trials showed a reduction in edge crossings.

There was a significant main effect on interaction mode, $F(1,15),\ p < 0.05$. The physics based interaction produced layouts with significantly fewer edge crossings than the normal interaction mode, with mean change in edge crossings of 0.578 (SE 0.087) compared to 2.811 (SE 0.538) for the normal interaction mode. Graph coloring did not significantly affect edge crossings, and no significant interaction between interaction mode and graph coloring was found.

5.3 Questionnaire Results

The results of the questionnaire comparing the physics interaction to normal interaction are summarised in Fig. 4. Significant results ($p < .05$) were found for questions 1, 3, and 4, with participants giving higher scores for the physics condition in all three questions. Participants overwhelmingly preferred the physics interaction, with 87.5% of participants choosing physics as the preferred mode. Participants also responded favourably to the graph coloring. Results for Q5 were 71.69 (SD 18.095), and Q6 68.19 (SD 15.753).

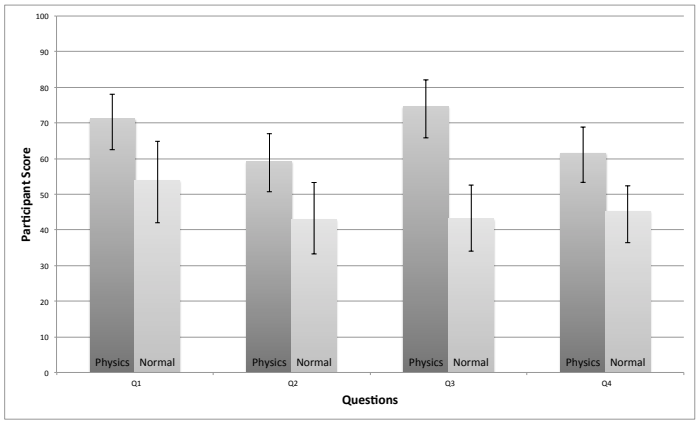


Fig. 4. Questionnaire Results for physics and normal interaction modes. Error bars show 95% CI

6 Discussion

The results of the user study show that the GION interaction technique is better in the untangling task than existing interactive methods. Specifically, hypotheses H1, H2, H5, and H9 were confirmed in the experiment. An interesting result is that users moved the mouse less in the GION condition, but made more mouse clicks. This suggests that participants spent more time fine-tuning the layout. Another unexpected result is that while GION produced layouts with lower stress and fewer edge crossings than the normal interaction, those values were worse than for the starting graph layouts. Using human interaction to untangle a large graph raises the question of what actually makes a good layout in this context. In particular, classical metrics like stress or edge crossings do not take into account the dynamics of the process and the mental map that the user creates during interaction. Intermediate layouts might not be good with respect to such metrics, but the user's interactive operations to create them can increase the users insight in the graph structure, and their value may depend on the preceding untangling process. Trying to measure such effects is however beyond the scope of this paper.

We deliberately restricted the user interface for the sake of a robust evaluation. For example, participants were not able to zoom or pan the graph drawing. A more complete system would also allow users to temporarily disable the physics engine in order to precisely control a single cluster. Multiple levels of clustering would also improve the technique, by allowing users to switch from coarse to fine grained interaction.

7 Implementation Details

Our video-wall consists of six NEC NP510W projectors each with a resolution of 1280x800 arranged in a 3x2 configuration. A camera based technique, as described by Raskar et al. [15], is used for geometric calibration of the projectors and for producing blending masks for smooth transitions between projectors. The computer used

in this work consists of 2x Quad Core Xeon processors, 12GB of RAM, and 2x Nvidia Geforce GTX 780 GPUs.

The system presented in this paper consists of a custom built application written in C++ with OpenGL for rendering. The physics simulation was developed using Box2D[3], an open source 2D physics library popular for game development. The Open Graph Drawing Framework (OGDF)[4] is used to provide graph data structures used by the application, saving and loading graphs at runtime, as well as for layout metrics. Vertex data is stored in an OpenGL Vertex Buffer Object (VBO). Edge data references the VBO and is rendered using an Index array. This reduces the amount of data transferred to the graphics card each frame and improves rendering performance by reducing the number of draw calls needed. Further enhancements were needed to improve rendering times on on a multi-projector display. A naive approach would simply involve rendering the graph in its entirety for each projector. Instead, we use a deferred rendering technique. The entire display is first rendered to an off-screen Framebuffer Object (FBO). Following this step, a portion of the FBO is rendered to each projector. This approach scales much better as the number of projectors increases, as the cost of each projector is just a single textured polygon.

8 Conclusion

In this paper we presented GION, a new interactive graph layout technique of large graph structures. GION is based on a physics engine to provide smooth and understandable animations to update the graph layout while the user moves a cluster. The results of a user study comparing GION with moving a single cluster at a time found the use of physics engine produced graphs with less stress, fewer edge crossings, and less mouse movement. Participants preferred the GION technique to moving a single cluster during the experiment.

We applied two standard quality layout metrics: stress and crossings. With GION, users constructed graph layouts that did not show significantly less stress or significantly fewer edge crossings, in comparison with the Fruchterman-Reingold algorithm. These results from our experiments lead us to question the validity of these two standard metrics for large graphs in the context of human layout improvement, and our work raises the question as to what quality metrics should be applied instead. We conjecture that measures like the precision of neighborhood preservation [5] will be better suited in this context than standard metrics for small graphs.

Acknowledgment This work was supported in part by a grant from the Australian Research Council - Discovery Grant DP120100248, Linkage Grant H2814 A4421, Tom Sawyer Software, and NewtonGreen Technologies.

[3] `http://box2d.org`
[4] `http://www.ogdf.net`

References

1. Ball, R., North, C., Bowman, D.: Move to improve: promoting physical navigation to increase user performance with large displays. In: Proc. of the SIGCHI conference on Human factors in computing systems. pp. 191–200. ACM (2007)
2. Catherine, R., Sudarshan, S.: Graph clustering for keyword search. In: Chawla, S., Karlapalem, K., Pudi, V. (eds.) COMAD. Computer Society of India (2009)
3. Dwyer, T., Lee, B., Fisher, D., Quinn, K.I., Isenberg, P., Robertson, G.G., North, C.: A comparison of user-generated and automatic graph layouts. IEEE Trans. Vis. Comput. Graph. 15(6), 961–968 (2009)
4. Gansner, E.R., Hu, Y., Krishnan, S.: Coast: A convex optimization approach to stress-based embedding. In: Wismath, S., Wolff, A. (eds.) Graph Drawing. LNCS, vol. 8242, pp. 268–279. Springer (2013)
5. Gansner, E.R., Hu, Y., North, S.C.: A maxent-stress model for graph layout. IEEE Trans. Vis. Comput. Graph. 19(6), 927–940 (2013)
6. Hachul, S., Jünger, M.: Large-graph layout algorithms at work: An experimental study. J. Graph Algorithms Appl. 11(2), 345–369 (2007)
7. Huang, W., Eades, P., Hong, S.H., Lin, C.C.: Improving force-directed graph drawings by making compromises between aesthetics. In: VL/HCC. pp. 176–183 (2010)
8. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In: Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques. pp. 165–172. SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
9. Merrick, D., Dwyer, T.: Skeletal animation for the exploration of graphs. In: Australasian Symposium on Information Visualisation, InVis.au. pp. 61–70. Christchurch, New Zealand (2004)
10. Murray, C., Merrick, D., Takatsuka, M.: Graph interaction through force-based skeletal animation. In: Australasian Symposium on Information Visualisation, InVis.au. pp. 81–90. Christchurch, New Zealand (2004)
11. Novák, P., Neumann, P., Macas, J.: Graph-based clustering and characterization of repetitive sequences in next-generation sequencing data. BMC Bioinformatics 11, 378 (2010)
12. Peck, S., North, C., Bowman, D.: A multiscale interaction technique for large, high-resolution displays. In: IEEE Symposium on 3D User Interfaces, 2009. 3DUI 2009. pp. 31–38. IEEE (2009)
13. Purchase, H.C.: Which aesthetic has the greatest effect on human understanding? In: Battista, G.D. (ed.) Graph Drawing. LNCS, vol. 1353, pp. 248–261. Springer (1997)
14. Purchase, H.C.: Metrics for graph drawing aesthetics. Journal of Visual Languages & Computing 13(5), 501–516 (2002)
15. Raskar, R., Brown, M.S., Yang, R., Chen, W.C., Welch, G., Towles, H., Seales, B., Fuchs, H.: Multi-projector displays using camera-based registration. In: Proc. of the conference on Visualization '99: celebrating ten years. pp. 161–168. IEEE Computer Society Press, San Francisco, CA, USA (1999)
16. Thomas, B.H., Calder, P.: Applying cartoon animation techniques to graphical user interfaces. ACM Transactions on Computer-Human Interaction (TOCHI) 8(3), 198–222 (2001)
17. Welman, C.: Inverse kinematics and geometric contraints for articulated figure movement. Master's thesis, Simon Fraser University (1993)
18. W.Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal 49, 291307 (1970)