Using Eye-Tracking as Interactive Input Enhances Graph Visualization

Mershack Okoe , Sayeed Safayet Alam, and Radu Jianu

Florida International University

Abstract— Much of the visual analysis of a graph reduces to a set of building-block visual tasks such as node scanning or edge and path tracing. These tasks may be trivial in small graphs but increase in complexity the larger and denser a graph visualization becomes. In this work, we use eye-tracking data as a real-time input to alter a graph visualization interactively and support its analysis. Specifically, we display labels of fixated nodes, we highlight edges as they are visually traced, and we dim out edges that pass through a users view-focus while having their endpoints far outside of it. We conducted a small informal user study to compare the performance of our eye-tracking enabled graph visualization versus a graph visualization system that only uses mouse input. The gaze-enabled visualization performed better in terms of accuracy and response time and was preferred by all participants.

Index Terms—Eye-tracking, gaze-contigent graph visualization.

1 Introduction

The novelty of our work lies in the use of eye-tracking data in real-time to support data-reading tasks in graph visualizations. Lee et al [7] showed that most graph-reading tasks can be reduced to a series of low-level tasks. These in turn reduce to node-label scanning and edge tracing. While such low-level tasks may seem trivial in small uncluttered graphs they are often time consuming in large, dense graphs. Here we show that we can leverage the benefits of eye-tracking as an interactive input to alter a graph visualization based on users' view-focus and thus improve the speed of both node-label scanning and edge tracing. While eye-tracking as an input has been explored in the human computer interaction community [2], it has not yet been, to the best of our knowledge, applied to data visualization.

Specifically, we use eye tracking data to achieve two things. First, we reduce clutter using a fovea-based filtering that dims edges that pass through a users view-focus but have their endpoint far removed. Second, we highlight edges as they are visually traced. Third, we display labels of fixated nodes. All these interactive visual responses are subtle and gradual to avoid a sense of over response known as the Midas touch problem [5] . Our intention was to develop an eye-tracking enabled visualization where gaze is used as a proxy for user intention rather than as a control input. Our work falls thus in the realm of gaze-contingent applications and attentive interfaces [2, 8].

We conducted a small informal user study to compare the performance of our gaze-enabled graph visualization to a graph visualization system that uses only mouse interactivity. The results show that our system performed better in terms of time and accuracy and was better liked by tested participants.

2 Related Work

Eye tracking has been used extensively in psychology, cognitive sciences, neuroscience, and computer science research for offline diagnosis of people's visual attention patterns [2]. However, it was also explored as an interactive input into visual interfaces. Duchowski divides such gaze-input systems into selective and gaze-contingent [2]. Selective systems use gaze-input in a way similar to a mouse to control an interface. While eye-tracking should presumably be faster than the mouse, selective systems have achieved mixed results and low adoption rates because of eye-tracking disadvantages: low accuracy and the inability to distinguish between looking and controlling, also known as the Midas Touch problem [5, 2]. Alternatively, gaze-contingent systems change the display as unobtrusively as possible to provide more informative details based on a users gaze. Examples of gaze-contingent work includes one that drives a story narration based on what holds a user's interest [1] and another that provides help on words that confuse a user in an electronic document [4]. It is currently accepted that eyes are not control organs and shouldn't be treated as such. Instead they should be used as a proxy for user intention in the context of visual attentive interfaces [8]. In line with this approach, we provide contextual graph information in response to a user's gaze.

To the best of our knowledge, in graph visualization eye tracking has only been used in a diagnostic role, as exemplified by Huang et al [3], and not as an interactive medium.

3 Methods

To develop a gaze responsive graph visualization, we focused our methods on two issues: improving gaze accuracy and providing interactive visual responses.

To achieve the first we applied Kumar's real-time fixation smoothing technique on our raw eye-tracking data [6]. Still, we discovered that often there was an offset from the real gaze position to gaze coordinates provided by the eye tracker. To address this we used the known graph topology. We assumed that long fixations generally had to match network node positions since users would not fixate empty space, and edge fixations as part of edge tracing should be short. We thus matched offset-vectors between subsequent long fixations to offset-vectors between nodes lying close to these fixations (Fig. 1). Combining this information with traditional distance-to-target thresholds led to better node fixation accuracy.

For the second, we have implemented the following types of gaze prompted visual responses: label display for fixated nodes (Fig. 2), highlighting of gaze-traced edges and separation of overlapping such edges (Fig. 2), and fovea based edge-filtering (Fig. 3). Labels are displayed in a semi-transparent, unobtrusive color. Gaze-traced edges are computed by a formula combining the amount of edge recently scanned, how long intermediate edge points were fixated, and whether any of its endpoints was recently fixated. For longer than average edges, the algorithm is currently able to highlight edges as they are traced. If two viewed edges are overlapping, they are curved away from each other in a smooth animation. Finally, a fovea-based edge filtering dims edges that pass through the users fovea but have both endpoints outside of it. We use two circles centered at the user's focus

- *email: mokoe001@cis.fiu.edu.*
- *email: salam011@cis.fiu.edu.*
- *email: rdjianu@cis.fiu.edu.*

point to gradually fade the filtering effect between the areas in which it is applied (foveated region) and where it is not (peripheral). The specific formula used to compute edge transparency was:

$$EdgeAlpha = f(pd) + (1 - f(pd)) \times min(1, \frac{d}{R_2}),$$
$$\text{where } f(pd) = 1 - \frac{pd - R_1}{R_2 - R_1}.$$

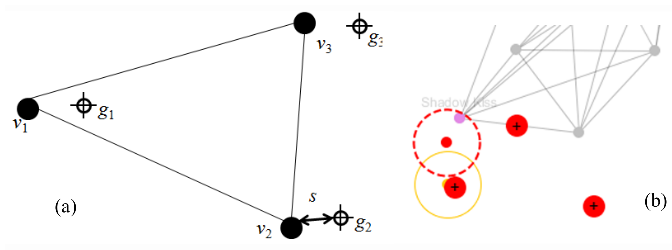

Fig. 1. (a) Correcting the gaze input: Even though fixations g1,g2,g3 are not exactly over graph vertices v1,v2,v3 their relative position matches that of the proximal graph vertices. We therefore conclude that g1,g2,g3 were fixations on the graph vertices v1,v2,v3. (b) Example of a gaze correction: the corrected input shown as a dotted red circle is is closer to the graph node than the original gaze-input (yellow circle).

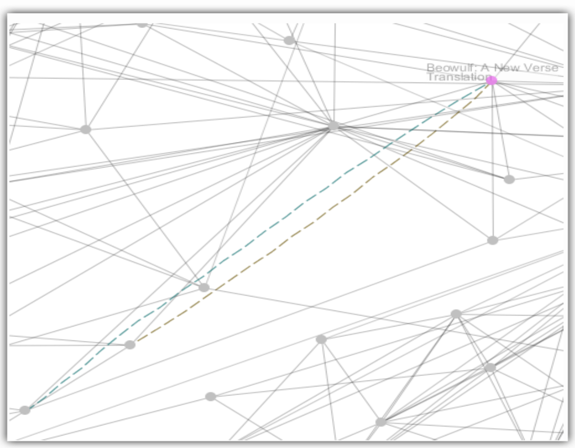

Fig. 2. Node labeling and visual edge tracing. Semi-transparent node labels are shown for fixated nodes. Visually traced edges are highlighted (dotted lines) and separated if they overlap.

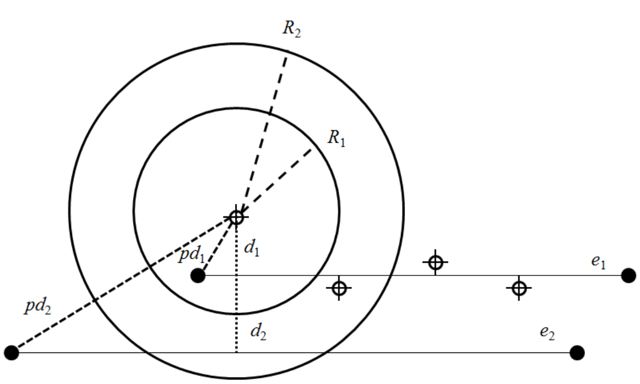

Fig. 3. Foveated edge filtering.

4 Evaluation

We evaluated our gaze-enabled graph visualization (eye-tracking condition) against a mouse-only version of itself (control condition) in a small, informal, within-subjects study with eight users. We used a book recommendation network dataset and tasks based on the graph task taxonomy by Lee et al [7]. Specifically, subjects were asked to perform three tasks: (1) neighbor task: determine whether two given nodes are directly connected (Y/N); (2) path task: determine whether 3 given nodes form a path (Y/N); (3) label task: determine the number of neighbors of a highlighted node that start with a given letter. Half of the subjects performed the tasks first in the eye-tracking condition and then in the control condition, the other half in reverse order. We measured task completion time and accuracy. After the study, subjects also selected their preferred visualization and rated the eye-tracking visualization using a 5-point Likert scale.

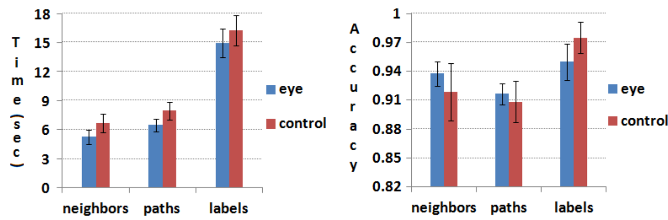

Fig. 4. Response time and accuracy bar charts for the three tasks

As shown in Fig.4, the eye-tracking visualization performed better in response time for all tasks. It was also more accurate for the neighbor and path tasks but not in the label task. All users rated the eye-tracking visualization as a four on the Likert scale and preferred it over the traditional visualization.

5 Conclusion

We have described a gaze-enabled graph visualization and an evaluation of its effectiveness over a mouse-only graph visualization. The contribution of this work is three fold: first, we have shown that eye tracking can be used as an input medium to improve the accuracy and response time of graph reading tasks in large dense graphs; second, we have introduced several gaze-prompted visual responses in the context of graph visualization; third we have introduced a novel method of handling the inaccuracies of measured gaze points by using the graph structure to interpret the semantic of a users gaze.

References

[1] R. A. Bolt. A gaze-responsive self-disclosing display. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 3–10. ACM, 1990.

[2] A. T. Duchowski. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, 34(4):455–470, 2002.

[3] W. Huang, P. Eades, and S.-H. Hong. Beyond time and error: a cognitive approach to the evaluation of graph drawings. In *Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization*, page 3. ACM, 2008.

[4] A. Hyrskykari, P. Majaranta, and K.-J. Räihä. Proactive response to eye movements. In *INTERACT*, volume 3, pages 129–136, 2003.

[5] R. J. Jacob and K. S. Karn. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind*, 2(3):4, 2003.

[6] M. Kumar, J. Klingner, R. Puranik, T. Winograd, and A. Paepcke. Improving the accuracy of gaze input for interaction. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 65–68. ACM, 2008.

[7] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization*, pages 1–5. ACM, 2006.

[8] T. Selker. Visual attentive interfaces. *BT Technology Journal*, 22(4):146–150, 2004.