A Set of Multi-touch Graph Interaction Techniques

Sebastian Schmidt[1,2] sebaschm@st.ovgu.de
Miguel A. Nacenta[1] manacent@ucalgary.ca
Raimund Dachselt[2] dachselt@ovgu.de
Sheelagh Carpendale[1] sheelagh@ucalgary.ca

[1] InnoVis Group, Interactions Lab
Department of Computer Science
University of Calgary, Canada

[2] User Interface & Software Engineering Group
Department of Simulation and Graphics
University of Magdeburg, Germany

ABSTRACT
Interactive node-link diagrams are useful for describing and exploring data relationships in many domains such as network analysis and transportation planning. We describe a multi-touch interaction technique set (IT set) that focuses on edge interactions for node-link diagrams. The set includes five techniques (TouchPlucking, TouchPinning, TouchStrumming, TouchBundling and PushLens) and provides the flexibility to combine them in either sequential or simultaneous actions in order to address edge congestion.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

Keywords: Multi-touch, Interaction Techniques, Gestures, Node-link Graph Manipulation, Interaction Technique Set

INTRODUCTION
In this research we focus on redesigning node-link graph exploration interfaces for use with multi-touch input. Node-link graphs are critical in many domains such as the visualization of communication and social networks, and transportation planning. However, they raise specific interaction challenges; for example, congestion of nodes and links [23], crossing of links [19], and overview and detail [8]. The use of multi-touch input to address these challenges is promising because the increase in available contact points may support more significant edge detangling and because multi-point input has been shown to enable better collaborative analysis [13]. By applying multi-touch capabilities to graph manipulations we extend the possibilities of graph interaction through combinations of simple actions.

We designed an interaction technique set (IT set) that supports a broad range of interactive analysis styles for node-link graphs, with a special emphasis on the manipulation of edges. We created a group of interaction techniques by redesigning some existing interactive techniques (EdgePlucking [22], EdgeLens [23], and pinning) and combining these with interactive variations of algorithmic edge techniques such as edge bundling [12], and a new technique inspired by the human perception of motion [1]. The design of the IT set is driven by the need to provide flexible and powerful techniques to enable collaborative analysis of graphs, but also to make a system that can reliably interpret simultaneous gestures by one or several people, and enhance capabilities by enabling combinations of gestures.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ITS'10, November 7-10, 2010, Saarbrücken, Germany.
Copyright 2010 ACM 978-1-4503-0399-6/10/11...$10.00.

RELATED WORK
Two main areas of prior work relate to this research: graph visualization, and multi-touch interaction technique sets.

Interactive Node-link Graph Visualization. Graph layouts have been often applied to real-world data and tasks. Due to the scale and complexity of real world data, these layouts tend to be dense and often contain difficult to read edge configurations [11]. Much previous work on graph layouts has focused on algorithmic approaches to create readable layouts [2], and on the significance of issues such as edge crossings and bends in layout aesthetics [18].

Interactive approaches that try to mitigate problems such as edge congestion include zoom and local magnification techniques [4, 10, 20]. These approaches tend to work on the graph as a whole or focus on the nodes, despite the fact that edges had been identified as one of the central aesthetic issues. Recent interaction advances that deal directly with edges in connected diagrams include Wong et al.'s EdgeLens [23] and EdgePlucking [22], Holten's edge bundling [12] and Moscovich et al.'s link-based navigation techniques [16] all of which serve as inspiration and base for this work.

Multi-touch Interaction Technique Sets. Multi-touch input is an active area of research with a history of several decades [6]; several studies have shown that enabling multi-touch can be beneficial for interaction [17, 7]. However, most relevant for this work are the recent advances in the development of multi-touch gesture sets. Although based on a large amount of work on sketch recognition and free-hand gestures [15, 3], it was only during the last few years that HCI research has addressed the construction of gesture sets for multi-touch applications. Wu et al. identify the creation of multi-touch gesture sets as a relevant problem, and provide guidelines to build effective sets [24]. Wobbrock et al. also address the creation of gesture sets, this time through user elicitation [21], which has also been applied to the creation of diagrams [9]. Our work draws from previous research on multi-touch gesture sets but applies it to edge-based node-link manipulation.

IT SET DESIGN RATIONALE

Designing a set of multi-touch interaction techniques is significantly different from designing individual techniques. In our design we strove to balance desirable properties of interaction techniques from the general HCI literature (e.g., power, memorability, discoverability and performance) with a choice of gestures that make their recognition feasible and reliable. Achieving distinguishability is one of the prominent challenges of multi-touch IT set design because not only can gestures be composed of several simultaneous touches, but also several separate gestures (by the same or other people) can take place simultaneously, making it possible for the recognition algorithm to group the different touches in the wrong way. To achieve these goals, and to enable the combination of different techniques into meaningful composite actions, we used a wide range of gesture types (e.g., tapping, tracing, and crossing gestures), leveraged gesture temporal characteristics (e.g., thresholds), and exploited geometrical relationships between gestures (e.g., parallel movements).

AN IT SET FOR GRAPH EXPLORATION

This section describes our edge manipulation IT set. Note that the focus of the set is on analysis, and therefore we do not provide techniques for the creation of node-link graphs. Our starting point is Lee et al.'s task taxonomy for graph visualization analysis [14]. Specifically, we predominantly support topology-based tasks such as finding out which nodes are adjacent to which nodes, or finding which path between two nodes is shortest.

TouchPlucking

TOUCHPLUCKING is a simple yet powerful technique that can manipulate the trajectory of one or more edges. TouchPlucking is based on EdgePlucking, a mouse-cursor technique designed for the desktop [22], which we extend to benefit from two-handed touch interaction. TouchPlucking allows one or more edges to be "grabbed" so that the edge(s) change their trajectory while staying connected to the two nodes that it links (see Fig. 1). The plucked edge(s) behave like flexible strings tied at both ends. This technique can reduce clutter by removing edges from the focus of attention and can also help reveal which nodes are connected to which edges.

We provide several alternative ways to initiate TouchPlucking. Touching an edge directly and dragging it activates the selected edge curving it from its nodes through the current position of the touch for as long as the finger is in contact with the surface (see Fig. 1 a). The initial touch area of the finger might be in contact with several edges, in which case all of them will be plucked. Alternatively, a touch that starts on blank space and drags across one or several edges will "capture" all the crossed edges (Fig. 1 b). This plucking method facilitates the selection of large sets of edges.

We also introduce a mechanism to facilitate plucking of specific edges that are located in congested areas. In these cases the fat finger problem makes it difficult to select only one edge at first touch. We achieve single edge selection by touching a node and holding it for 1.5 seconds, which causes all edges adjacent to this node to straighten out and change their color. When starting a second touch on these edges, none is plucked right away, but tracing along an edge highlights the edge with the direction most similar to the trace. This highlighted edge can now be plucked with the second touch and the first touch can be released (Fig. 2).

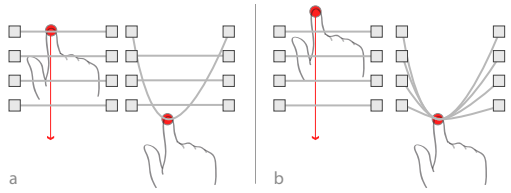

Figure 1: Two variants of TouchPlucking. (a) a single edge is selected and plucked (b) a drag starting on a blank space plucks all edges crossed.

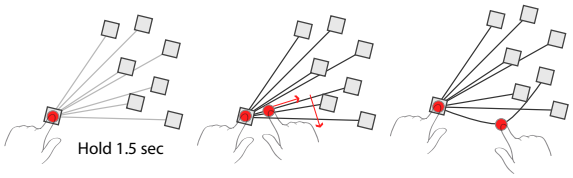

Figure 2: Plucking of a single edge out of a congested area using two touches. The node is held for 1.5s, which straightens out all the node's edges. A second touch drags along the edge to select the edge, after which normal plucking resumes.

Regardless of how TouchPlucking is started, lifting the finger from the surface will cause the edges animate to their original positions. The transient nature of the technique allows for exploration without a permanent modification of the state of the graph. Separate TouchPlucking actions can be merged into one by bringing two or more plucking touches within a close distance of each other. When this condition is detected and one touch is released, all the edges that this touch had plucked are transferred to the other touch (Fig. 3). The effect is to toss an edge into an existing group of edges.

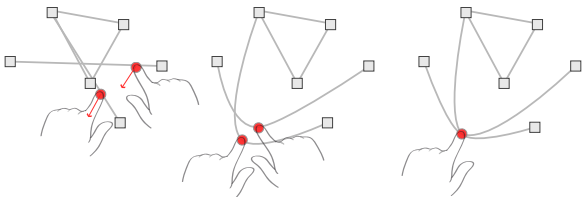

Figure 3: Bringing two active plucking touches close to each other merges them.

TouchPinning

Sometimes it is necessary to change the edge layout for a longer period of time in order to keep the clarity that has been gained or to make room for other exploration tasks. We therefore introduce *TouchPinning* as a means of fixing already curved edges using the metaphor of a push-pin. After having selected an arbitrary number of edges through TouchPlucking, holding the finger for 1.5 seconds at a desired location will place a 'pin' and keep the edges curved after the touch has been lifted. To visually indicate this, a small push-pin symbol appears (Fig. 4).

Pinned edges can also be turned into a pluck by starting a touch-drag on the push-pin. Tapping on the push-pin releases all of its pinned edges, which smoothly animate back to their

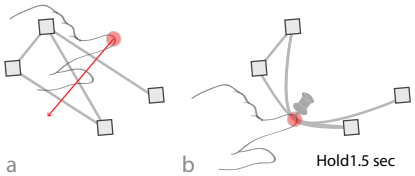

Figure 4: While plucking (a), a bundle of edges can be pinned by holding the touch for more than 1.5 s (b).

unbent state. It is also possible to add further edges to an existing pin by plucking other edges and moving them close to the pin position. When released, the new edges are added to the already pinned ones. Visual feedback (a connecting line) indicates whether the newly plucked edges will be added to the push-pin or released.

Pinning is useful when people want to pluck a larger number of edges at different positions and keep this state, or if they have less fingers available than necessary for plucking several edges. It allows the creation of a clear view on the changed layout without hand occlusion.

TouchStrumming

TOUCHSTRUMMING takes advantage of the saliency of human motion-perception to offer an alternative way to visualize the connection of one or more edges. TouchStrumming is similar to plucking, but uses a short "flick" gesture that triggers the vibration of an edge for some time (Fig. 5) The underlying metaphor is a guitar string showing a strumming motion. The amplitude of the strumming motion is proportional to the length of the gesture.

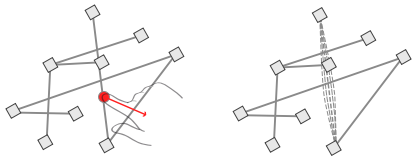

Figure 5: Using a short flick gesture to strum an edge.

Strumming can also be activated by a flick gesture on a node, which triggers vibration of all edges connected to this node (Fig. 6). Using this technique it is easy to spot all nodes which are connected to the selected one. As with some of the introduced techniques, this one can be parameterized. For example, the vibration can propagate with decreasing amplitude to the edges connected to second-level nodes, etc.

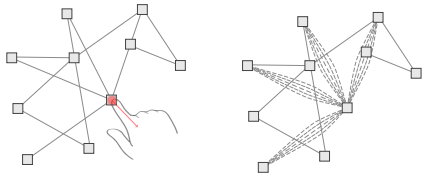

Figure 6: The same gesture as in figure 5 can be used on nodes to strum all connected edges.

One could argue that color highlighting of the affected nodes would be sufficient for the same tasks. However, we deliberately designed our techniques to not consume valuable visual attributes which will likely be used to encode other properties. This is in line with the work of Bartram and Ware [1] who investigated the benefits of motion patterns to emphasize data objects in a static visualization. TouchStrumming is a lightweight way of focusing on edges compared to plucking and pinning and can facilitate awareness by other participants of a collaborative analysis, e.g., on a tabletop.

TouchBundling

With TOUCHBUNDLING we contribute a technique that allows analysts to bring together the paths of several edges to avoid cluttering or to highlight relationships between groups of nodes. This technique provides interactive control of bundling instead of automatically bundling edges through an algorithm[12].

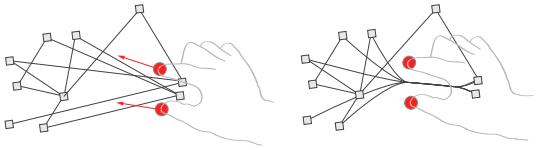

Figure 7: Performing a funnel gesture on edges clusters all edges which are oriented in the direction of the gesture movement and bundles them.

To create a bundle, we use a simple two-finger funneling gesture where the touches enclose the edges that need to be bundled (Fig. 7). The two touch points are moved into the same direction to elongate the bundled length or to collect more edges. TouchBundling will collect all edges that cross the imaginary line between the touches in a similar direction as both touches move. If a node comes in between the two touches, all edges connected to it are added to the bundle. The length of the bundling segment is determined by the start and the end of the gesture, performed by both fingers.

Bundles stay when the gesture is finished, and can later be altered in shape and location by dragging the ends of the bundling segment. As with TouchPinning, edges assigned to the bundle can be added and removed using TouchPlucking. Since the edge bundle can be considered an (aggregated) edge itself, it can also be plucked like a single edge. Bundles can be deleted by tapping anywhere on the bundling segment.

PushLens

A PUSHLENS is an interface object with a round shape. Edges connected to nodes contained within the area of the lens are not altered, whereas the rest of the edges are bent away from the lens (see Fig. 8). The PushLens is based on a long tradition of focus+context and lens techniques [20, 5] but its closest relative, EdgeLens [23] was proposed for mouse interaction.

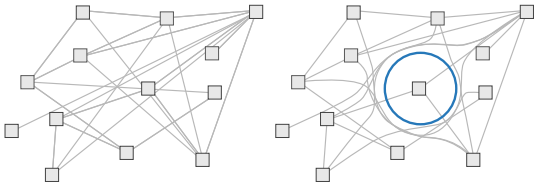

Figure 8: Assigning a PushLens on a graph visualization to reduce congestion of edges.

PushLenses can be used to interactively define clutter-free areas. If a lens contains a single node it makes easy to see its connecting edges and their directions. If a group of nodes is enclosed, the lens will highlight the relationships between those nodes, and will exclude edges that do not connect with

any of the internal nodes. PushLenses can also reveal information that is displayed under the node-link graph (e.g., the geographic map under a flight graph).The lens alters the trajectory of edges and allows all the other techniques to be applied within its area, as long as the other techniques do not start their touches on the rim of the lens.

A PushLens is manipulated by a group of different interaction techniques. The lens is created through a three-finger interaction technique in which all three touches are near in space and time. Once it is created, the lens is represented by a circular outline. The lens can be moved by a single touch on its outline, and translated and scaled through a pinching interaction technique that starts on two points of the outline or rim. The lens disappears if it is tapped.

DISCUSSION AND CONCLUSION

In this paper we present a new multi-touch interaction technique set designed for the exploration of node-link diagrams. The set was designed to allow analysts to perform well-known tasks for the domain, but it puts special emphasis on the relatively ignored aspect of edge manipulation. Some techniques that form the set adapt single-point techniques to multi-point interfaces, others are entirely new.

Our technique set was designed to enable the exploration of node-link diagrams through a powerful, memorable, discoverable and efficient combination of techniques; furthermore, the techniques are designed to avoid distinguishability problems that arise in multi-touch, multi-user sets when touches from separate gestures and/or people are interpreted as belonging to one gesture and vice versa.

Also it is our experience that the combination and simultaneous use of these techniques with multi-touch input broadens the interaction possibilities with respect to sequentially constrained single-point interfaces. For example, our interface allows one to separate irrelevant edges with one hand and simultaneously perform an operation on the remaining set of edges with the other hand (e.g., strumming or bundling). This operation feels natural and straightforward with this set, but would require planning and a longer sequence of operations with a single-point interface. While promising in its ability to offer combinations variations both sequentially and simultaneously, this IT set will need to be tested in real world situations with real world tasks and challenges.

ACKNOWLEDGMENTS

We would like to thank NSERC, SMART Technologies Inc, iCORE, and CFI for research support.

REFERENCES

1. L. Bartram and C. Ware. Filtering and brushing with motion. *INFOVIS '02*, pp. 66–79, 2002.
2. G. Battista, P. Eades, R. Tamassia, and I.G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, 1998.
3. T. Baudel and M. Beaudouin-Lafon. Charade: remote control of objects using free-hand gestures. *Commun. ACM*, pp. 28–35, 1993.
4. B. B. Bederson and J. D. Hollun. Pad++: A zooming graphical interface for exploring alternate interface physics. In *UIST '94*, pp. 17–26, 1994.
5. E. A. Bier, M. C. Stone, K Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: the see-through interface. In *SIGGRAPH '93*, pp. 73–80, 1993.
6. W. Buxton. Multi-touch systems that i have known and loved. http://billbuxton.com/multitouchOverview.html.
7. W. Buxton and B. Myers. A study in two-handed input. *CHI Bull.*, pp. 321–326, 1986.
8. A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *Comput. Surv.*, pp. 1–31, 2008.
9. M. Frisch, J. Heydekorn, and R. Dachselt. Diagram editing on interactive displays using multi-touch and pen gestures. In *Diagrams '10*, pp. 182–196, 2010.
10. G. W. Furnas. Generalized fisheye views. *CHI '86*, pp. 16–23, 1986.
11. I. Herman, G Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *TVCG*, pp. 24–43, 2000.
12. D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *TVCG*, pp. 741–748, 2006.
13. P. Isenberg, A. Bezerianos, N. Henry, S. Carpendale, and J-D. Fekete. Coconuttrix: Collaborative retrofitting for information visualization. *CG and A*, pp. 44–57, 2009.
14. B. Lee, C. Plaisant, C. S. Parr, J-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *BELIV '06*, pp. 1–5, 2006.
15. A. C. Long, Jr., J. A. Landay, and L. A. Rowe. Implications for a gesture design tool. In *CHI '99*, pp. 40–47, 1999.
16. T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J-D. Fekete. Topology-aware navigation in large networks. In *CHI '09*, pp. 2319–2328, 2009.
17. T. Moscovich and J. F. Hughes. Indirect mappings of multi-touch input using one and two hands. In *CHI '08*, pp. 1275–1284, 2008.
18. H. C. Purchase. Which aesthetic has the greatest effect on human understanding? In *GD '97*, pp. 248–261, 1997.
19. H. C. Purchase, D. Carrington, and J. Allder. Evaluating graph drawing aesthetics: defining and exploring a new empirical research area. *Computer Graphics and Multimedia.*, pp. 145–178, 2004.
20. M. Sarkar and M. H. Brown. Graphical fisheye views of graphs. In *CHI '92*, pp. 83–91, 1992.
21. J. O. Wobbrock, M.R. Morris, and A.D. Wilson. User-defined gestures for surface computing. In *CHI '09*, pp. 1083–1092, 2009.
22. N. Wong and S. Carpendale. Supporting interactive graph exploration with edge plucking. *VDA '07*, 2007.
23. N. Wong, S. Carpendale, and S. Greenberg. Edgelens: An interactive method for managing edge congestion in graphs. *INFOVIS '03*, pp. 7–15, 2003.
24. M. Wu, C. Shen, K. Ryall, C. Forlines, and R. Balakrishnan. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In *TABLETOP '06*, pp. 185–192, 2006.