VisWall: Visual Data Exploration Using Direct Combination on Large Touch Displays

Mallika Agarwal[*] Arjun Srinivasan[†] John Stasko[‡]

School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA

Abstract

An increasing number of data visualization tools are being designed for touch-based devices ranging from smartwatches to large wall-sized displays. While most of these tools have focused on exploring novel techniques to manually specify visualizations, recent touch-based visualization systems have begun to explore interface and interaction techniques for attribute-based visualization recommendations as a way to aid users (particularly novices) during data exploration. Advancing this line of work, we present a visualization system, VisWall, that enables visual data exploration in both single user and co-located collaborative settings on large touch displays. Coupling the concepts of direct combination and derivable visualizations, VisWall enables rapid construction of multivariate visualizations using attributes of previously created visualizations. By blending visualization recommendations and naturalistic interactions, VisWall seeks to help users visually explore their data by allowing them to focus more on aspects of the data (particularly, data attributes) rather than specifying and reconfiguring visualizations. We discuss the design, interaction techniques, and operations employed by VisWall along with a scenario of how these can be used to facilitate various tasks during visual data exploration.

Index Terms: Human-centered computing—Visualization;

1 Introduction and Background

More recently, there has been an increased interest within the visualization community to explore tools on devices where conventional input modalities such as keyboard and mouse are not available [18]. One input modality, in particular, that has gained growing attention is touch input. Several systems and studies have investigated visualizations on both large (wall or tabletop) [19, 20, 23, 24] and small (tablets or phones) [5, 13, 17, 27–29] touch displays. In addition to illustrating the potential benefits and challenges of touch input and varying display sizes, these efforts have also shown that designing visualization systems on new devices often imposes different constraints and requires significant changes in the interaction style and the system's interface [13, 20, 29].

Most existing touch-based visualization tools allow creating and interacting with visualizations as a means to explore data. To create visualizations on these tools, however, users are typically expected to manually specify (via specific pen/touch gestures or control panel widgets) both the the visualizations they want to create and the mappings between attributes and visual encodings (e.g., color, size). An alternative approach presented by desktop-based tools like Voyager [35] and Show Me [21] is that of automatically recommending visualizations to users based on their selection of data attributes. Prior research has shown that suggesting visualizations and providing users with alternative data representations can alter decision making strategies [16] and help users validate their findings [30]. However, even with these potential benefits, only a few touch-based visualization systems have examined designs to facilitate similar attribute-based visualization recommendations for data exploration.

For instance, PanoramicData [37] lets people get univariate summary visualizations by dragging attributes from a list or out of existing visualizations. However, to create visualizations with two or more attributes, users still need to explicitly specify a visualization type and manually configure it using the available attributes. On the other hand, TouchPivot [13] enables a tight coupling between a data table and the visualization, and recommends visualizations to users based on their column selections in the data table. The interactions in TouchPivot were primarily designed for a tablet device, however, and may not be effective in a larger display setting that supports different affordances and imposes additional challenges with respect to the placement and access to objects on the screen. In our work, we explore an alternative approach to let users create multivariate visualizations on large touch displays while still primarily focusing on data attributes (as opposed to explicitly specifying visualization types or mapping attributes to particular visual encodings).

Blending the concepts of *direct combination* [10, 11] and *derivable visualizations* [6, 37], we explore an approach that enables rapid construction of visualizations using attributes of previously created visualizations. Building upon existing work on visualization recommendation engines [34, 35] and touch-based visualization systems [13, 20, 37], we employ this approach in a tool, *VisWall*. Similar to existing systems [13, 35, 37], VisWall lets users create univariate visualizations by directly selecting available data attributes. However, once two or more visualizations are created, users can "merge" them to create a new visualization based on the attributes of the merged visualizations. Complementing this merge operation with the ability to navigate alternative visualizations, copy, and split visualizations, VisWall allows people to visually explore their data by focusing more on aspects of the data (in this case, data attributes) rather than specifying and reconfiguring visualizations. With an initial prototype, we designed and refined the system through iterative discussions and feedback from research colleagues based on preliminary testing of alternative designs.

We describe VisWall's current design focusing on our key design goals and the operations supported by the system. We also present a usage scenario to illustrate how VisWall can enable a fluid interaction experience [7] for visual data exploration on large touch displays.

2 VisWall

2.1 Design Goals

Three high-level goals drove the design of the current interface:

DG1. Accommodate users with varying levels of visualization expertise. Since users of a large wall-sized display can have varied backgrounds and expertise (e.g., executive decision makers, expert data analysts), it was important that the design supported both experts as well as novice and intermediate-level users of visualization tools. Along these lines, we highly resonated with the six design guidelines prescribed by Grammel et al. [8]. In particular, we focused on two specific guidelines they proposed: *suggesting visualizations* (**DG1a**) and *supporting iterative specification* (**DG1b**).

[*]email: mallika.agarwal@gatech.edu
[†]email: arjun010@gatech.edu
[‡]email: stasko@cc.gatech.edu

IEEE VIS 2019: Short Papers
20-25 October, Vancouver, BC, Canada
978-1-7281-4941-7/19/$31.00 ©2019 IEEE

DG2. Allow users to leverage available space. One key difference between a large touch display and desktop/laptop settings is the space afforded by the device (both physical and on-screen). Prior work [1, 2, 15, 32] has also shown that spatially organizing information lets users represent relationships between items and mentally create complex structures like clusters. Hence, we wanted to ensure that the design allowed users to freely move around in the physical space and yet interact with all components, and navigate and arrange items freely within the interface.

DG3. Support both individual and collaborative use. Although collaborative data exploration is not the primary focus of our work, considering the target display size, co-located use of the tool is a very natural scenario to expect [2, 12, 33]. Accordingly, it was important that the design and interactions not only accommodated one user at a time but also provided basic support for co-located visual data exploration. The current version of VisWall was designed and tested on a 84” Microsoft Surface Hub. To limit the scope of our current work and keeping in mind the dimensions of the Surface Hub, we based our design on the assumption that a maximum of two people would engage with the tool at the same time. With this assumption in mind, we considered designs that let both users have access to common interface elements (**DG3a**) and helped avoid redundancy during co-located exploration by making users aware of visualizations that are present elsewhere on the canvas (**DG3b**).

2.2 User Interface

Figure 1 highlights VisWall’s user interface which contains two primary components: *attribute panels* and the main *visualization canvas*. Three collapsible attribute panels (Figure 1A,B,C) can be accessed from the left, top, or right sides of the screen. These panels contain the names of the available attributes in the loaded dataset and indicate the attribute type: quantitative (**#**), categorical (**A**), or temporal (📅). Having access to attribute panels on both sides and the top facilitates access to attributes regardless of the user’s standing position relative to the screen (**DG2**). Furthermore, this repetition also allows personalized or shared access to attributes when two people are using the tool in a co-located exploration setting (**DG3a**). The main visualization canvas (Figure 1D) is an unbounded pan and zoom enabled 2D canvas that allows free-form arrangement of visualizations (**DG2**).

Visualizations are rendered within ***tiles*** (Figure 5). Tiles can be freely dragged around, re-sized, and placed anywhere on the canvas. Tiles can be created directly using attributes from the attribute panels or by combining two existing tiles. Attributes and/or tiles used to create a tile are converted to ***bricks*** that appear at the bottom of a tile (Figure 5D).

2.3 Usage Scenario

To provide a sense of how VisWall functions, we first present a hypothetical usage scenario. Imagine Sarah, a movie producer who has various scripts to choose from and wants to make a data-driven decision on which movie to invest in. To assist this process, she downloads a dataset of movies released in 2016 with 12 attributes for each movie including *budget*, *primary_genre*, *gross*, *content_rating*, *director_name*, among others. Sarah wants to explore this dataset to understand the type of movie she should produce, answering questions such as which movie genres have the best return on investment, and if there is a content rating that has proven more profitable than others. To visually explore the available dataset of movies, Sarah loads it into VisWall, which displays the list of available attributes in the attribute panel (Figure 1A).

To begin her exploration, Sarah scans the attributes and, to see the typical investments in movies, simply drags the *budget* attribute onto the canvas. As she drags the attribute, the system automatically creates a tile with a *strip plot* showing the distribution of *budget* values across all movies (Figure 2). Looking at the strip plot, Sarah

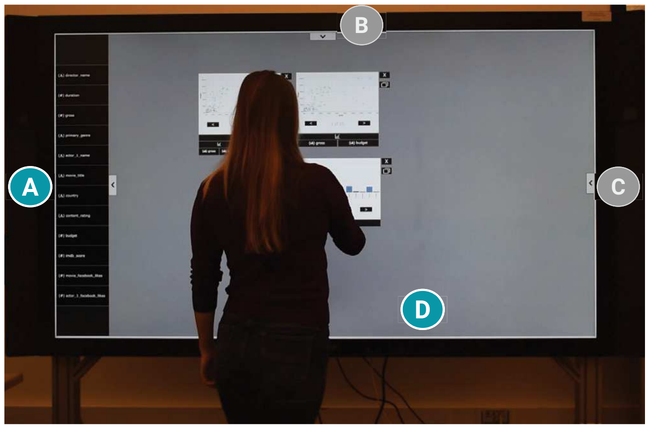

Figure 1: A user exploring the movies dataset on VisWall running on an 84” Microsoft Surface Hub. (A,B,C) Attribute panels and (D) Visualization Canvas. In this case, only one of the attribute panels (A) is open.

wonders if there is a better visualization that will give her a more aggregated overview of the *budget* values. Sarah taps on the line of text under the strip plot that reads “*1 of 9*”. This opens a window that gives thumbnail previews of alternative visualizations showing *budget* values (Figure 3). Sarah notices a *histogram* and taps it to replace the strip plot. Using this histogram, Sarah notes that more than half the movies in 2016 were produced with a *budget* of less than *50M*. Sarah then follows the same steps to create a histogram of *gross* values which shows that most movies earned between *0-150M*.

To understand if there is a correlation between the amount of money invested (*budget*) and the amount of money a movie makes (*gross*), Sarah simply drags the tile showing the histogram of *gross* values onto the tile showing the histogram of *budget* values. This

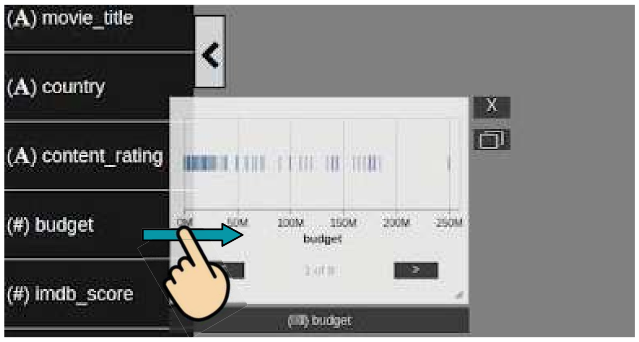

Figure 2: Dragging attribute *budget* from the attribute panel automatically creates a tile with a strip plot.

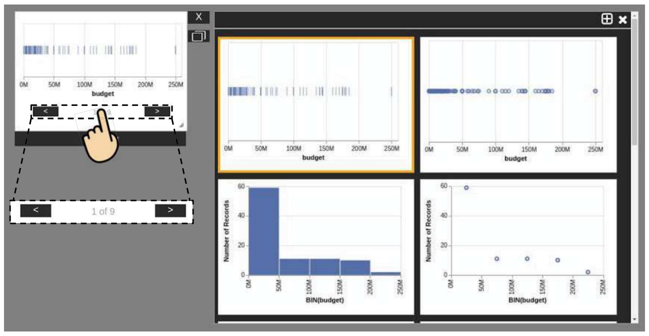

Figure 3: Thumbnail-based browsing of alternative visualizations.

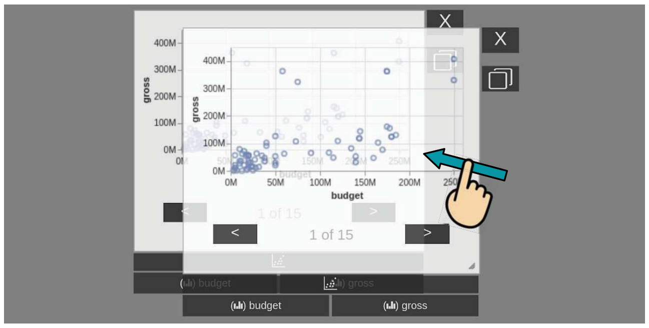

Figure 4: Preview shown before merging *gross* and *budget*.

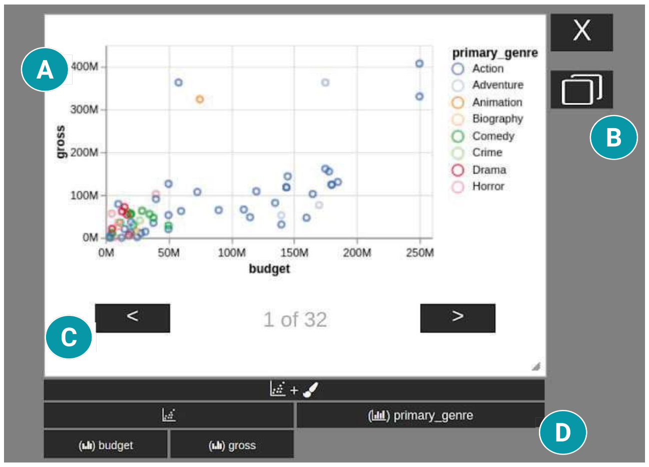

Figure 5: Tile showing a colored scatterplot created by merging the *budget vs. gross* scatterplot with a bar chart of *primary_genre*. (A) Visualization, (B) quick-access buttons, (C) visualization browsing row, and (D) bricks showing attributes and visualizations that were merged to create the current tile.

replaces the tiles showing histograms for *budget* and *gross* with a single new tile showing a *scatterplot* of *budget vs. gross* (Figure 4). Sarah notices that while there is fairly positive correlation between the two attributes, there are some high-profit movies in the *budget* range of *50-100M*, and similarly some other movies that had a *budget* of around *150M* and incurred a substantial loss .

Keeping the scatterplot tile on the canvas, Sarah drags in the *primary_genre* attribute which shows a *bar chart* of available genres and number of movies for each genre. Sarah drags this tile directly onto the existing scatterplot of *budget vs. gross* which updates it to a colored scatterplot as shown in Figure 5. Using this chart, Sarah notices that three of the five highest grossing movies were primarily *action* movies, whereas *primary_genres* for the other two movies were *adventure* or *animation* . To explore the relationship between *budget* and *gross* across another dimension, Sarah decides she now wants to consider the *content_rating* attribute. Correspondingly, Sarah simply copies the colored scatterplot tile showing attributes *budget*, *gross*, and *primary_genre* using a hold-and-drag gesture (Figure 6).

Next, Sarah removes the *primary_genre* attribute from the scatterplot by simply dragging out the brick corresponding to *primary_genre* at the bottom of the tile (Figure 7). This reverts the tiles back to a regular non-colored scatterplot between *budget* and *gross*, and a bar chart of *primary_genre*. Sarah then drags in the *content_rating* attribute from the attribute panel and merges it with the *budget vs. gross* scatterplot to create a colored scatterplot once again. Sarah notes observations such as most low-budget, low-grossing movies are *'R'* rated, there is only one *'R'* rated movie grossing

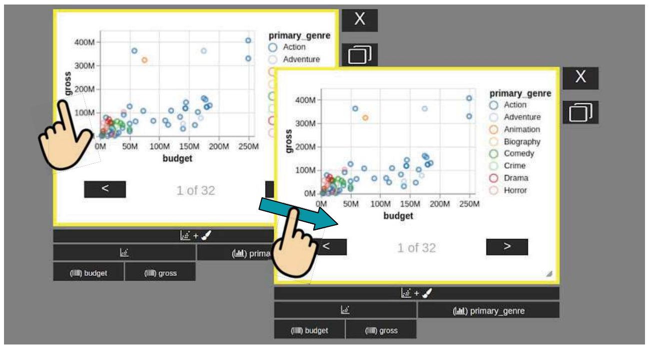

Figure 6: Tile copied using a hold-and-drag gesture. In this case, a yellow stroke is added to highlight duplicate tiles.

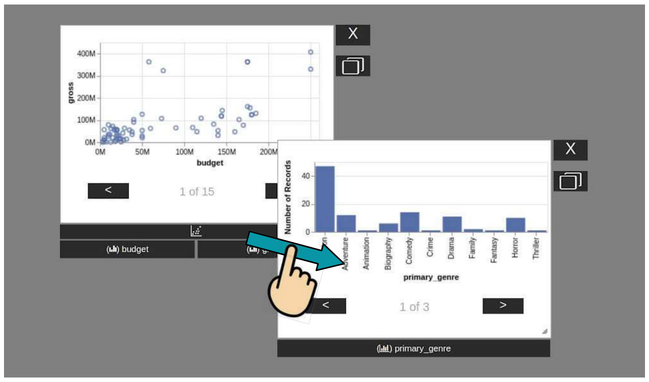

Figure 7: Tile shown in Figure 5 is split by dragging the *primary_genre* attribute brick.

over *100M*, and so on. She then places the two colored scatterplots (one colored by *primary_genre* and another by *content_rating*) next to each other so she can compare the two to identify more trends. Deciding she wants to consider other attributes and return to these visualizations later, Sarah pans the canvas to move the existing tiles out of focus and drags in new attributes, continuing her exploration.

2.4 Operations

VisWall provides a set of six core operations that support visual data exploration scenarios similar to the one illustrated above. Below we list these operations along with a brief description of the interactions used to perform them.

Add. Users can create visualizations by dragging them from the attribute panel onto the canvas. As soon as an attribute is dragged, a tile with a default visualization (e.g., a histogram for numeric attributes) is rendered under the dragging finger (Figure 2).

Merge. Visualizations with two or more attributes can be created by merging existing tiles. To merge tiles, users can drag a tile close to the top-left corner of another tile. When there is sufficient overlap, the system also provides a "preview" of the resulting visualization (Figure 4). Note that because the goal was to support attribute-focused visual data exploration (**DG1**), VisWall's merge operation creates a new visualization based on the attributes of the visualizations that were merged. This differs from previous visualization systems that determine the result of a merge operation based on the orientation of charts [3, 4, 17, 20] or intersection of set values [26].

Once a merge occurs, an icicle plot like visualization at the bottom of a tile displays the steps performed to create a tile (Figure 5D). The tile that was dragged onto another tile is always added to the right subtree. For example, through the bricks in Figure 5D, we can interpret that the colored scatterplot was created by merging a scatterplot of *budget* and *gross* with a bar chart of *primary_genre*.

Navigate Alternative Visualizations. VisWall also recommends alternative visualizations for each tile. To navigate through these alternatives, people can use the navigation buttons or tap on the text under a visualization in a tile (Figure 5C) to preview and choose from thumbnails of alternative visualizations (Figure 3).

Copy. To avoid repeated construction of existing visualizations, VisWall also lets users create copies of existing tiles. Users can perform a bimanual hold-and-drag gesture [9] (Figure 6) or use the clone button (⧉) to copy a tile (Figure 5B). When a tile is copied or whenever two tiles showing the same visualization for the same set of attributes exist on the canvas, a stroke of the same color is applied to the tiles allowing identification of duplicates (Figure 6). This is particularly useful in collaborative settings when two people may be using different parts of the canvas individually [12] (**DG3b**).

Split. Supporting iterative specification (**DG1b**) and allowing users to rapidly "deconstruct" or revert back to a previous visualization, VisWall also supports splitting a tile back into the tiles that were merged to create it. Users can simply drag a brick from the bottom of a tile to split it (Figure 7). When a tile is split, the system automatically combines the remaining bricks of the tile. For example, if *gross* was removed from the tile in Figure 5, *budget* and *primary_genre* would be automatically merged.

To create a tile from an existing tile's brick without splitting it, one can also perform a bimanual hold-and-drag gesture by placing a finger on the tile while dragging out the desired brick (Figure 8).

Remove. To remove a tile from the canvas, users can tap the delete icon on the top-right corner of a tile (Figure 5B).

2.5 Implementation

VisWall is implemented as a web application running in a standard browser. We use the browser's default mechanisms to detect all active touches on the screen at any given point and then process the input streams to support the various operations listed above. We use Vega-lite [31] to specify and render the visualizations.

We use Compass [35] to generate a ranked list of possible visualizations given a set of data attributes. The CompassQL [34] query used is configured to consider only four mark types: bar, point, line, and tick. The transforms configured include sum, average, binning, and count for the applicable attribute types. The encoding channels considered by Compass in our configuration include size, shape, color, and faceting.

We further prune the results from Compass to remove visualizations that are differently oriented variations of each other. For visualizations created directly from an attribute (e.g. Figure 2), we use the default visualization ranking applied by Compass. However, for visualizations created by merging two tiles, we modify the ranking order returned by Compass so that the suggested visualizations are more consistent with the visualizations that were merged. For instance, in Figure 5, even though there are other possible visualizations such as a colored strip plot, charts with aggregated XY-axis values, etc., a colored scatterplot is shown first because it maintains the *budget vs. gross* scatterplot.

Since the recommendations are generated based on a tile's attributes, they contain various combinations of the attributes. In other words, among the possible list of visualizations for the tile in Figure 5 would also be charts that show *primary_genre* along with *gross* or *budget* on the XY-axis and are colored or sized by *budget* or *gross* . However, depending on the visualizations being merged, such charts might be ranked lower than others.

3 Limitations and Future Work

Through initial testing with potential users, we have identified several areas for improvement within the system and opportunities for future research. VisWall currently only supports touch-based input. While users can still use a pen, it is treated the same as a touch

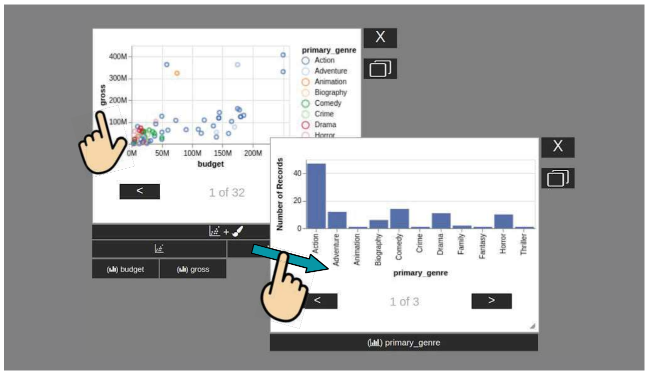

Figure 8: Tile for *primary_genre* copied via an attribute brick using a hold-and-drag gesture.

input. However, preserving the current set of interactions for touch, we can envision further incorporating pen input to perform tasks like brushing-and-linking [37] or adding annotations [19, 25]. An additional input modality like the pen would also open up avenues for enhanced interactions like stapling [9] to support other tasks such as workspace management (e.g., creating groups of similar or related tiles, or moving many tiles together).

An advantage of direct combination as an interaction technique is that it can be used for letting people perform multiple operations on a pair of objects. We currently support only one operation via direct combination: merging. However, one can envision other options such as "compare", "link", etc. being provided when two tiles are combined. Enabling these operations and designing the corresponding interface components to present them during combination (e.g., using context menus as shown in Holland and Oppenheim's initial demonstration [11]) remains an open area for future work. Predictability is a known challenge with direct combination-based interaction [22] (e.g., one may expect to see a stacked bar chart when combining two bar charts of categorical attributes but the system may generate a small multiples chart instead). Thus, conducting formal user studies to understand potential advantages and limitations of the presented technique is an immediate next step to evaluate VisWall's usability and utility in real-world data analysis scenarios.

As highlighted by Wongsuphasawat et al. [36], there are tradeoffs between browsing- and specification-based strategies for visual data exploration. An area for future work is to consider how we can strike a balance between automatically creating visualizations within tiles and allowing users to reconfigure tiles. In future versions of the tool, we can also extend the recommendation ranking function to consider parameters like the size of the tile (e.g. faceted charts could be ranked higher when the tile is large) and potentially even analyze the active canvas and data distribution when ranking visualizations [8]. Lastly, we can also incorporate ranking functions from tools such as Graphscape [14] to better maintain consistency between visualizations before and after merging.

4 Conclusion

We presented VisWall, a system that facilitates visual data exploration using direct combination and derivable visualizations on large touch displays in both single user and co-located collaborative settings. We discussed the key design goals we followed while developing the system and provided system details including the user interface and operations supported. Through an example scenario of use, we sought to illustrate VisWall's features and potential to enable a naturalistic, fluid visual data exploration experience.

Acknowledgments

This work was supported in part by the National Science Foundation grant IIS-1717111.

References

[1] C. Andrews, A. Endert, and C. North. Space to think: large high-resolution displays for sensemaking. In *Proceedings ACM CHI '10*, pp. 55–64, 2010.

[2] S. K. Badam, F. Amini, N. Elmqvist, and P. Irani. Supporting visual exploration for multiple users in large display environments. In *Proceedings of IEEE VAST*, pp. 1–10. IEEE, 2016.

[3] J. H. Claessen and J. J. Van Wijk. Flexible linked axes for multivariate data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2310–2316, 2011.

[4] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott. Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation. In *Proceedings of ACM UIST*, pp. 71–83. ACM, 2017.

[5] S. M. Drucker, D. Fisher, R. Sadana, J. Herron, et al. Touchviz: a case study comparing two interfaces for data analytics on tablets. In *Proceedings of ACM CHI '13*, pp. 2301–2310, 2013.

[6] C. Dunne, N. Henry Riche, B. Lee, R. Metoyer, and G. Robertson. Graphtrail: Analyzing large multivariate, heterogeneous networks while supporting exploration history. In *Proceedings of ACM CHI '12*, pp. 1663–1672, 2012.

[7] N. Elmqvist, A. V. Moere, H.-C. Jetter, D. Cernea, H. Reiterer, and T. Jankun-Kelly. Fluid interaction for information visualization. *Information Visualization*, 10(4):327–340, Oct. 2011.

[8] L. Grammel, M. Tory, and M.-A. Storey. How information visualization novices construct visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):943–952, 2010.

[9] K. Hinckley, K. Yatani, M. Pahud, N. Coddington, J. Rodenhouse, A. Wilson, H. Benko, and B. Buxton. Pen+ touch= new tools. In *Proceedings of ACM UIST '10*, pp. 27–36, 2010.

[10] S. Holland, D. R. Morse, and H. Gedenryd. Direct combination: A new user interaction principle for mobile and ubiquitous hci. In *Proceedings of International Conference on Mobile Human-Computer Interaction*, pp. 108–122. Springer, 2002.

[11] S. Holland and D. Oppenheim. Direct combination. In *Proceedings of ACM CHI '99*, pp. 262–269, 1999.

[12] P. Isenberg and D. Fisher. Collaborative brushing and linking for co-located visual analytics of document collections. *Computer Graphics Forum*, 28(3):1031–1038, 2009.

[13] J. Jo, S. L'Yi, B. Lee, and J. Seo. Touchpivot: Blending wimp & post-wimp interfaces for data exploration on tablet devices. In *Proceedings of ACM CHI '17*, pp. 2660–2671, 2017.

[14] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer. Graphscape: A model for automated reasoning about visualization similarity and sequencing. In *Proceedings of ACM CHI '17*, pp. 2628–2638, 2017.

[15] D. Kirsh. The intelligent use of space. *Artificial intelligence*, 73(1-2):31–68, 1995.

[16] D. N. Kleinmuntz and D. A. Schkade. Information displays and decision processes. *Psychological Science*, 4(4):221–227, 1993.

[17] R. Langner, T. Horak, and R. Dachselt. Vistiles: Coordinating and combining co-located mobile devices for visual data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):626–636, 2018.

[18] B. Lee, P. Isenberg, N. H. Riche, and S. Carpendale. Beyond mouse and keyboard: Expanding design considerations for information visualization interactions. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2689–2698, 2012.

[19] B. Lee, R. H. Kazi, and G. Smith. Sketchstory: Telling more engaging stories with data through freeform sketching. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2416–2425, 2013.

[20] B. Lee, G. Smith, N. H. Riche, A. Karlson, and S. Carpendale. Sketchinsight: Natural data exploration on interactive whiteboards leveraging pen and touch interaction. In *Proceedings of IEEE PacificVis '15*, pp. 199–206, 2015.

[21] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007.

[22] B. Myers, S. E. Hudson, and R. Pausch. Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction*, 7(1):3–28, 2000.

[23] T. Nagel, L. Pschetz, M. Stefaner, M. Halkia, and B. Müller. mæve–an interactive tabletop installation for exploring background information in exhibitions. In *Proceedings of the International Conference on Human-Computer Interaction*, pp. 483–491, 2009.

[24] C. North, T. Dwyer, B. Lee, D. Fisher, P. Isenberg, G. Robertson, and K. Inkpen. Understanding multi-touch manipulation for surface computing. In *Proceedings of INTERACT '09*, pp. 236–249, 2009.

[25] H. Romat, N. Riche, K. Hinckley, B. Lee, C. Appert, E. Pietriga, and C. Collins. Activeink:(th) inking with data. In *Proceedings of ACM CHI*, 2019.

[26] R. Sadana, T. Major, A. Dove, and J. Stasko. Onset: A visualization technique for large-scale binary set data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1993–2002, 2014.

[27] R. Sadana and J. Stasko. Designing and implementing an interactive scatterplot visualization for a tablet computer. In *Proceedings of AVI '14*, pp. 265–272, 2014.

[28] R. Sadana and J. Stasko. Designing multiple coordinated visualizations for tablets. In *Computer Graphics Forum*, vol. 35, pp. 261–270, 2016.

[29] R. Sadana and J. Stasko. Expanding selection for information visualization systems on tablet devices. In *Proceedings of ACM ISS '16*, pp. 149–158, 2016.

[30] P. Saraiya, C. North, and K. Duca. An insight-based methodology for evaluating bioinformatics visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):443–456, 2005.

[31] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, 2017.

[32] F. M. Shipman III, C. C. Marshall, and T. P. Moran. Finding and using implicit structure in human-organized spatial layouts of information. In *Proceedings of ACM CHI '95*, pp. 346–353, 1995.

[33] M. Tobiasz, P. Isenberg, and S. Carpendale. Lark: Coordinating co-located collaboration with information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1065–1072, 2009.

[34] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Towards a general-purpose query language for visualization recommendation. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, p. 4, 2016.

[35] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):649–658, 2016.

[36] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of ACM CHI '17*, pp. 2648–2659, 2017.

[37] E. Zgraggen, R. Zeleznik, and S. M. Drucker. PanoramicData: data analysis through pen & touch. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2112–2121, 2014.