Panoramic Interaction with Interval Data Based on the Slider Metaphor

Phillip C. S. R. Kilgore, Marjan Trutschl, Urška Cvek
Department of Computer Science
Louisiana State University in Shreveport
Shreveport, USA
e-mail: {pkilgore, mtrutsch, ucvek}@lsus.edu

***Abstract*—Users utilize information visualization tools to explore their multivariate data in two- and three-dimensional environments in order to uncover patterns, form hypotheses or pursue questions of interest; they filter, search, zoom, and interact with the data. We first present a short survey of interactive selection mechanisms for information visualization and related work. We follow with our panoramic range slider and show the examples of its application and utility and present results of a case study. Finally, we elucidate on the advantages that the panoramic range slider can provide while identifying areas of further refinement.**

Keywords-range selection; range slider; linear control; interval data interaction; query inversion; visualization interaction; graphical user interfaces.

I. Introduction

With the exponential increase in scale and amount of data generated follows the need for tools for dynamic and intuitive interaction. Users want to be able to formulate their strategies, pose questions and uncover patterns of interest, to identify information hidden in such data. Information visualization is an effective means of making sense of multivariate data. We typically utilize the two-dimensional (2D) or three-dimensional (3D) space to plot data records as dots, bars, lines and glyphs, mapping dimensions to position, size, shape, color and, in some cases, motion. Interactivity with these data increases the utilization of the human visual perception, providing for a more meaningful analysis of the data [1].

Dynamic queries using interactive tools are a powerful means of data interaction, due to their visual nature and support for incremental, rapid visual actions on the data. They can elucidate global trends, and help find answers to specific questions [2]. These queries are formulated using widgets such as sliders, buttons and check boxes. Dynamic HomeFinder [3] was one of the first systems that successfully integrated dynamic queries into visualization.

A variety of techniques are utilized for exploratory analysis of data sets. Users commonly seek an answer to the question ”What is the typical behavior of the system from which we are collecting data?”. In such cases, the user is interested in data which is bound by some range specified in terms of a central tendency and variance. However, users sometimes look for data that deviates significantly from the norm (outliers).

An example of this can be seen during analysis of differential gene expression data, where interesting records are not representative of the majority of the data. In this kind of analysis, two or more microarray samples are taken with the intent of discovering genes which show significant changes in regulation, and the underlying hypothesis is that these genes belong to one or more biological pathways [4]. The expectation is that comparatively few genes will show regulatory changes, so outliers typically become objects of intense study.

While differential gene expression experiments are a good example of an outlier-based approach, there are other instances where this sort of analysis is warranted. For instance, an educational institution focused on improving standardized test scores (either globally or within some demographic) might wish to approach the task by either determining steps taken by the top-scoring students, which typically do not represent the majority of test-takers. For instance, in 2009, the average composite score for the American College Testing assessment was 21.1, while participants receiving the maximum score of 36 represented 0.04307% of all participants [5]. Since college admission and scholarship awards are often based on scores received on such tests, it is within these institutions' best interest to analyze the behavior of the outliers to claim notable improvements in their own population.

Finally, there exists the possibility of interaction with a measure that is intrinsically modular. Interaction with a closed dimension (a finite yet unbounded dimension such as degree measure or clock time) is an example of such a phenomenon. In these cases, we are dealing with half-open intervals of the form $[0, n)$, where $xn \equiv 0, x \in \mathbb{Z}$.

We present a brief overview of dynamic interactive queries, focusing on the scrollbar and slider metaphors. We discuss related work and present our panoramic range slider that can be utilized for selection in both closed and open intervals. Additionally, we show examples of its application using real-life multivariate datasets and conclude with a brief case study. Finally, we postulate several use cases for the panoramic range slider and discuss future research directions.

II. Related Work

Several continuous-domain controls in computing are metaphors for analog devices. For instance, the *slider* concept shares much in common with linear potentiometers (or ”faders”) and its availability is nearly ubiquitous. Sliders are typically implemented as discrete controls; however, with sufficient granularity, they can be used to emulate continuous value selection. Another popular metaphor is the *dial* concept, which is a metaphor for a radial potentiometer and is a core control in the Qt GUI framework [6]. *Thumbwheels* (or rollers) are another radial form of continuous input that (while less commonly implemented) can be found in various GUI toolkits [7]–[9]. AlphaSlider [10] is a query device that allows users to select one item from an alphabetically arranged list without using a keyboard.

These metaphors are useful for input of a single, scalar value. However, range input is a more difficult task to approach because there are few metaphors for which a tangible analog instrument exists. A commonly provided form of range-based input includes the *scrollbar* concept, which can provide input specifying a range with a fixed width. The *piano roll* metaphor commonly found in music production software (such as FL Studio [11], Reason [12], and Rosegarden [13]) is continuous with respect to time and is based off of the recording media used for player pianos. The piano roll also has an interesting property: its ability to effectively visualize the content of the underlying document. Daschelt et al. provide a method of facet-based navigation called *FacetZoom* [14] based on a stack of one-dimensional treemaps that is suitable for navigating (amongst other things) textual data.

A. Interval-Based Interaction

The above range-based controls show weaknesses when attempting to manipulate numeric intervals. The piano roll metaphor may be useful for time-domain visualization and manipulation; however, these are vector data, and numeric intervals tend to be a subset of a scalar domain. Scrollbars allow for the positioning of ranges, but do not intrinsically permit selection of range width. Finally, FacetZoom [14] works best with a finite subset of its underlying domain, as it was designed to expedite navigation to a single datum rather than a range of data.

A range slider (Fig. 1) is a control used to specify continuous intervals. On the left is a minimum bound indicator a (Fig. 1.a), and on the right, the maximum bound indicator c (Fig. 1.c). The space occupied by the combination of the bound indicators and the thumb represents a subinterval of the interval $[0, 1]$ (Fig. 1.d). Suppose that the bounds indicators have a width of w_b, that the thumb has a width w_t, and that the bounding area has a width w. Let x_a denote the position of the left side of a, and x_c denote the right side of c. Then, the range slider represents the closed interval $[x_a/w, x_c/w]$, and $w_t = x_c - (x_a + 2w_b)$.

Range sliders share much in common with the scrollbar metaphor, although a notable difference arises out of its usage. Like a scrollbar, a range slider consists of a thumb (or slider, Fig. 1.b) that is free to move within a rectangular bounding area (Fig. 1.d). Unlike a scrollbar, the two buttons (the bound indicators) associated with a range slider are connected to the thumb and define its width. Common applications of range sliders focus on filtering a range of records. Becker [15] used a double-edged slider with upper and lower thresholds on network maps, Eick et al. [16] used a categorical slider on software displays for selecting an arbitrary subset from hundreds of time-ordered software modifications, and Ahlberg [17] used a suite of double-edged sliders for FilmFinder. Spotfire [18] was the first commercially successful information visualization tool based on traditional dynamic queries utilizing range sliders.

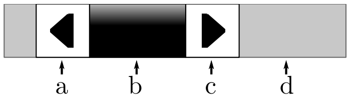


Figure 1. A range slider consists of several parts: a) minimum bound indicator, b) thumb, c) maximum bound indicator, and d) bounding region. The interval depicted is $[a, c]$.

The *pie of time* [19] is a circular range slider widget that allows selection of times of day. The inner circle has the full 24 hour cycle around its perimeter. The hours can be individually switched on or off and a range of hours can be clicked and dragged to enable/disable them. The *radial range selector* is a similar concept by Till Nagel [20], and gives the ability to select a range on any circular, repetitive event data, such as the time of day, weeks or seasons, and other circular data (such as angles, etc.).

Over the past couple of decades range sliders have been extended to spatial data to interactively select records within spatial proximity, such as the *TrapezoidBox* [21]. The TrapezoidBox interface allows the users to combine related query results using interactions. For medical data sets, Müller et al. utilized a *brushing histogram* view overlaid with a value range as a histogram, on which the user can make a selection [22]. A study of dynamic query sliders versus the brushing histograms was done by Li et al. [23] where they used both in an empirical experiment on a geographic data visualization tool. A user study led them to believe that the brushing histograms were superior for complex discovery tasks. A number of similar range-based selection mechanisms have been introduced, including parallel coordinate displays where each axis is a controlled attribute range, queries for temporal variations, peaks, valleys and slopes, and their selection includes not only the mouse and keyboard, but touch, gesture and speech. Readers are encouraged to visit [24] for a full taxonomy of interactive dynamic tools for visual analysis.

B. Problem Analysis

The range slider metaphor is applicable for both closed and open intervals. However, it may be useful to exclude a range, such that for all x, $x \notin [a, c]$, or equivalently, $x \in [0.0, a) \cup (c, 1.0]$. A naïve solution to this might involve linking a checkbox to the inverted interpretation of a traditional range slider, as seen in Fig. 2. When the control is used in a nested list of checkboxes, problems can occur. There, it is not clear what the checkbox affords: does the checkbox afford inversion, or (as an example) the domain's consideration for a dynamic query? It is also unclear what the checked state means even in the context where it is known to be associated with the slider.

This problem could (in part) be resolved by descriptive text provided by a label or tooltip; with a label, ambiguity is mitigated at the cost of increased usage of the display medium. Tooltips could also be used in this capacity (albeit with some introduction of delay) but this method does not lend itself well to environments which lack a mouse cursor (such as mobile devices). This control invites error, either in the form

Copyright (c) IARIA, 2014. ISBN: 978-1-61208-325-4

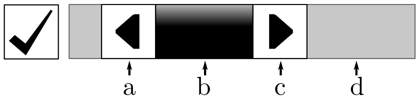


Figure 2. A possible incarnation of a range slider affording inversion by using a checkbox to toggle the inverted state. This method may not cascade well in nested lists involving other checkboxes.

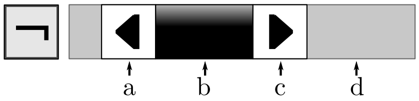
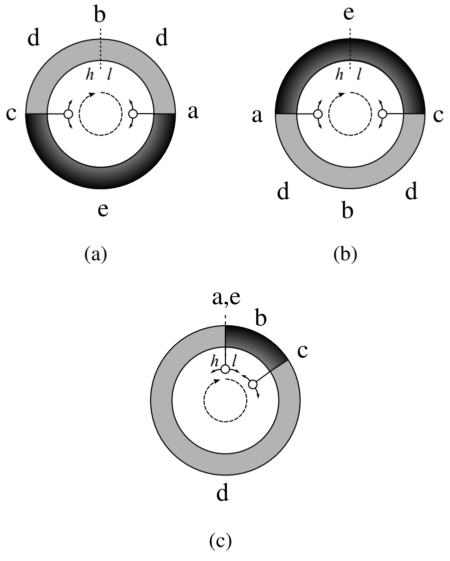


Figure 3. A slight improvement on the modal approach to range slider inversion is provided by replacing the checkbox with a push-button. The effect of the control is no longer ambiguous, but mode errors are still possible.



(a)

(b)

(c)

Figure 4. An augmented radial slider with a bound interface indicator. a) The interquartile range is selected. b) The inverse of (a), the extremes, is selected. c) A selection with an ambiguous lower bound because the domain boundaries (l, h) overlap.

of misinterpretation of the checkbox's function or in the form of mode error.

Recognizing that the checkbox is being used in a modal capacity, a possible improvement might be to replace the checkbox with a push-button depicting a glyph that explicitly labels the mode (Fig. 3). The button remains depressed during inversion; otherwise, it is raised. If an appropriate glyph is chosen, there is no longer any question as to what the control affords, and the mode of the slider can be discerned from the state of the button. This method also happens to violate the expected behavior of push-buttons in some environments (where they are monostable controls and the push state is expected to be transient) [25]–[27]. Some user interface (UI) toolkits do provide for bistable push-button controls [6], where it might be acceptable for a push-button to maintain a depressed state; however, while it allegedly improves upon the initial checkbox design, the construction presented in Fig. 3 still leaves the possibility for mode error.

A better alternative to prevent mode error would be removing the mode controlling inversion to begin with. One such example of this is the radial range selector presented by Nagel [20]. There are no modes implied in its design, and although no mechanism for range inversion is presented, it can be afforded by reversing the bound indicators. This control was designed with interaction of closed dimensions in mind, and in principle, it could be used with open dimensions as well.

However, there exist a number of issues which make it less useful in the latter capacity. Because it is designed to interact with cyclic data, there is no indication where the interface between the bounds (l and h) of an open dimension is. While intuition might tell us that this position is at 12:00, this point could (in principle) be anywhere, including the 3:00 position associated with a traditional Cartesian plot of $(x, y)_\theta = (\cos\theta, \sin\theta)$. Without any indication of this position, it would lead us to believe that the notion of a start position is arbitrary, but this is not the case with an open dimension, as they have a precise lower and upper bound.

This issue might be solved by explicitly marking this interface (as depicted in Fig. 4), but that introduces another problem. The presence of this boundary implies (by reasoning analogous to the interpretation of a clock face) that the lower (l) and upper (h) bounds are equivalent; again, this is not so with an open dimension. While this affords an inverted range where including the extremes would be desirable, it also affords inconsistent behavior of that interface.

Consider the use case where a radial range slider is used to filter property values. Any interval within the interquartile range is unambiguous: knowing that the interface represents the absolute bounds, we assume that the selection opposite to it must represent some range in the middle of the absolute range. It might be useful for the user to query values within the interquartile range of property values; such a selection is easily recognized because the semicircle opposite to the bounds indicator must indicate this half of the range (Fig. 4.a). Likewise, it might be desirable to look at the extremes in the data by inverting the selection; again, the semicircle intersecting the bound interface succinctly depicts this (Fig. 4.b).

When the selection sits on the threshold of the bound indicator, difficulty in interpretation can arise (Fig. 4.c). Here, a selection of the lowest value is probably intended, but it is questionable whether or not the highest value is also selected, because it shares the interface with the lowest value. This is a consequence of the metaphor. For a closed dimension, this is the expected and proper behavior: after all, $0° \equiv 360°$ in degree measure, and $0{:}00 \equiv 24{:}00$ on the face of a clock. Thus, regardless of the direction of the approach, the limit at any point is preserved, because the two values are (by definition) the same.

This is not the case with an open dimension. Here, the interface represents a jump discontinuity where the limits from either side are not equal; the interface (in fact) represents values with maximal distance. A reasonable interpretation of Fig. 4.c might conditionally represent either extreme based on the limit of the opposite direction to the handle, but this now overloads the behavior of the interface, because intersecting the interface indicator will represent two values.

III. Panoramic Range Sliders

While the radial range slider has issues, it has several appealing properties. Firstly, the selection range (with the expectation of the discontinuity at the bound interface) is clear, and this reduces the need for a mode to specify inversion. The lack of a mode is also appealing. If we could indicate for the open dimension that there was a definite, non-equal bound, we would solve one of the primary issues with the radial range slider. Thus, we utilize the linear depiction of the range.

One question that arises is how to visually depict inversion of the selected range. This can be shown by having the arrows on a classic range slider point inwards and to provide a thumb on both extremes of the bounding area (Fig. 5). The area represented by d is left unshaded to indicate that it is not part of the selection. We name the semantic extension of this metaphor a "panoramic range slider" due to its similarity to $360°$ panoramic photographs.

If one glues together the left and and right edges of the bounding area, an annulus is formed. Let τ be any point in $\mathbb{R}$ along the circumference of the outer rim, such that $\tau = 0$ denotes the point where the bounding area was joined, and let τ_l and τ_h respectively represent the positions of the lower and upper bounds. The mapping to the interval $[0.0, 1.0)$ can be calculated by the bijection $\tau(\theta) = \theta/2\pi$.

As with the traditional range slider, the resulting range is a subset $\mathcal{P}$ of $[0.0, 1.0]$. However, the circles are themselves modular in nature, such that any value of θ can be mapped to $[0.0, 2\pi)$ via the binary operator "$x \operatorname{fmod} y$". The fmod operator is surjective and is defined by (1). Thus, $\mathcal{P}$ can be defined as the set specified by (2).

$$x \operatorname{fmod} y \equiv x - y\lfloor x/y \rfloor \tag{1}$$

$$\mathcal{P} \equiv \forall\theta \in (\mathbb{R}|x_l < \theta < x_l + x_h), \exists x \in [0,1] : \tau(\theta \operatorname{fmod} 2\pi). \tag{2}$$

In this sense, the range slider as presented in Fig. 5 is somewhat like the ancient symbol Ouroboros, which depicts a snake eating its own tail. However, we find that if we straighten out this proverbial "snake" by ungluing the vertices at the new boundary, one may occasionally obtain two non-null subsets of $\mathcal{P}$: the set $[0, a)$ and the set $(c, 1]$. This happens when $x_a > x_c$, or when the minimum is greater than the maximum. We refer to this state as "negated state", such that the slider is interpreted to mean $x \notin [a, c]$.

It should be noted that the underlying dimension need not be intrinsically modular, just that it must be totally ordered and have definite bounds. By gluing the bounds together, the range slider is in principle the radial slider presented in Fig. 4. However, the linearization offers two new pieces of information. The range covered by the two handles is the range that will be selected; it is also apparent that there is a minimum and maximum in this range.

There are several ways that we suggest the slider could enter a negated state. One is to cross the bound indicators over another (e.g., moving the maximum indicator before the minimum indicator); another way is to move the thumb beyond the bounding region. Both of these methods have a severe drawback, as the user must carefully reposition the bound in an attempt to enter negated state, likely leaving an interval with different bounds and introducing error. Alternative mechanisms for triggering inversion are possible, however; in an environment for which a keyboard is present, the use of quasi-modal interaction (such as ctrl-click) can quickly and accurately result in the inversion. Other possible mechanisms include selection via an entry in a context menu.

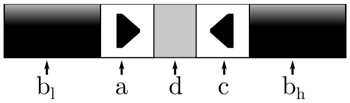


Figure 5. A panoramic range slider in negated state. The interval depicted is $[0.0, l) \cup (h, 1.0]$. A panoramic range slider is a linear realization of the radial range slider control in Fig. 4. Note that in its negated state, there are two handles (shown here as b_l and b_h).

IV. Evaluation

As a proof of concept, we have chosen to apply these techniques to a data set containing both open and closed dimensions. We generated a synthetic data set containing 10,000 randomly-generated samples of the tuple $(x \in \mathbb{Z}, y \in \mathbb{Z}, z \in \mathbb{Z}, r \in \mathbb{R}, \theta \in \mathbb{R})$. Each point is constrained within a maximum radius R and maximum height Z, such that $0 \le r \le R$ and $0 \le z \le Z$. Thus, a cylinder of radius R and height Z is a convex set encapsulating the data, and the points have Euclidean coordinates $(x = \lfloor r\cos\theta \rfloor, \lfloor r \sin\theta \rfloor, z)$. It should be noted that the data is devoid of semantics, and the task at hand is the selection of regions within that convex set.

The range sliders contribute to a 5-volume (the bounding volume) which consists of the bounds of four open dimensions and the closed dimension θ (Fig. 6). Intervals for r, θ and z can be combined to form a bounding volume specified by an annular sector prism. It is reasonable that r and z can be specified as open dimensions without any loss in usability, since these dimensions do not require modularity. A traditional range slider provides a proper subset of possible bounding prisms because it can hand intervals for θ such that $\theta_l \le \theta_h$. However, when $\theta_l > \theta_h$, two bounding prisms must be employed: one which that represents the interval $[0, \theta_h]$ and one that represents the interval $[\theta_l, 1.0]$.

The lack of modular behavior in the traditional range slider can interfere with the effective specification of the endpoints of the bounding prism. For instance, the disjoint nature of the two bounding prisms means that it is possible to accidentally specify two separate bounding prisms rather than the two intersecting ones. Secondly, in a complex data query, we are usually interested in the intersection of records which fit each constraint in the data query. However, a special exception to the semantics of the data query must be made to treat the two range sliders as a union and causes some difficulty in understanding the semantics of the query.

An alternative method is to duplicate each record, and to substitute θ with θ_{+360} in the duplicate. This method doubles the range of θ and therefore permits a single range slider to encapsulate the selected range; this results in data

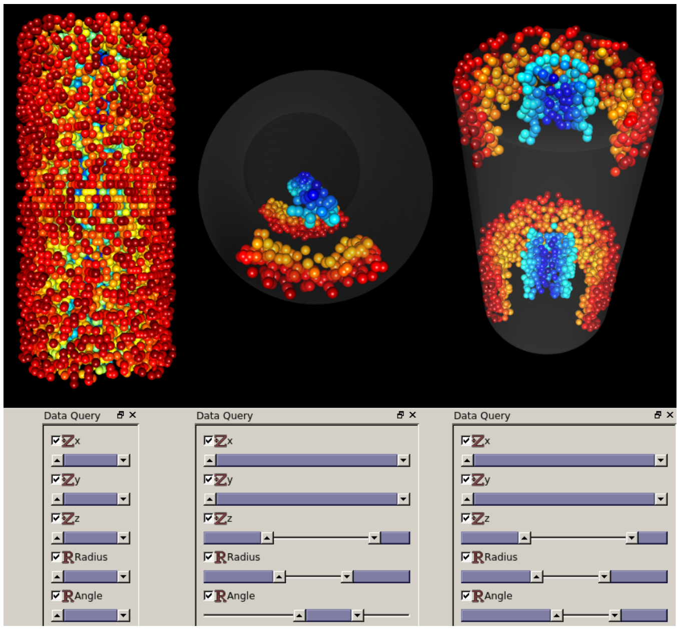


Figure 6. A collage of various selections for the above data set (records colored by radius). The second and third selections cannot be expressed by a single ordinary range slider per dimension.

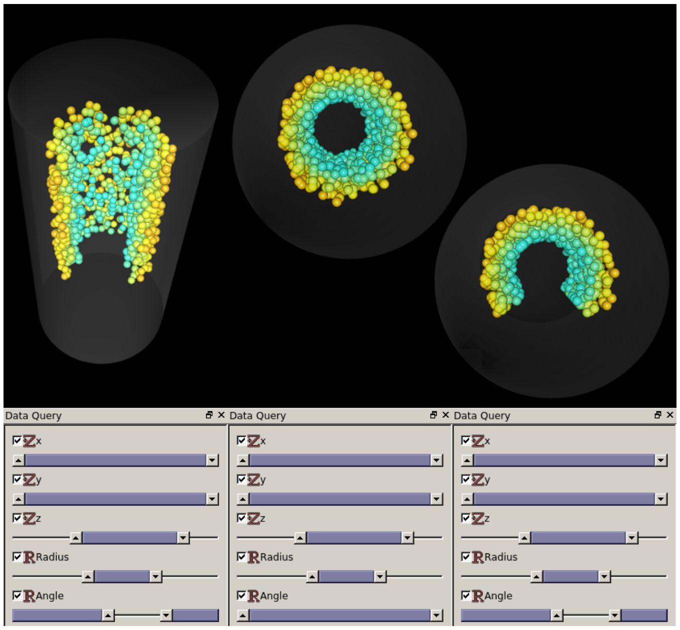


Figure 7. Various selection bounds on the data presented in Fig. 6 using only the three range sliders provided in the data query. The second angle dimension is no longer necessary and can be omitted from the data entirely.

redundancy, which may require infeasible amounts of CPU time and memory for large data sets.

Panoramic range sliders address this problem because the underlying abstraction involves a closed dimension. The query is simplified because the disjunction is not directly exposed to the user. A single panoramic range slider produces a single set which has exactly two bounds to be manipulated by the user (Fig. 6, Fig. 7). Thus, while it is reasonable to view the resulting set as the union $[0, \theta_h] \cup [\theta_l, 1]$, the modular nature of the control also means that the user views the set as a single interval $[\theta_l, \theta_h + 2\pi]$ when $\tau_l > \tau_h$.

Fig. 6 demonstrates the behavior of the panoramic range slider. The first selection captures the entire data set. The second selection demonstrates both traditional range slider behavior (applied to the angle) and sliders with negated state. The third selection is similar, but it negates the angle. Neither of these are afforded by the ordinary range slider, which requires the interval to be contiguous. Accurate inversion is afforded by a hotkey (Fig. 7).

In both Fig. 6 and Fig. 7, neither the dimension θ_{+360} nor data duplication is necessary when using a panoramic range slider. Because θ is inherently modular, it fits the concept of the panoramic range slider extremely well. Furthermore, the complement of the bounding volume is easily obtained by inverting both the z, r, and θ components (Fig. 6).

V. Conclusions and Future Work

The addition of modularity to range sliders provides greater control over how data is filtered in several ways. First, panoramic range sliders permit the concept of excluding data that falls within a given interval in addition to the inclusive approach taken by traditional range sliders. Secondly, it allows inherently closed dimensions to be represented in a way that reflects their modular nature. Finally, it preserves bounds, making it simpler to determine what the effective bounds represented by the data are.

The data set used during the evaluation has served as a viable proving ground for these properties for two reasons. The data set is sufficiently large: in the case of the provided data set, a high degree of occlusion can be seen such that interior records are difficult to investigate. The data set is also dense: in this data set, it is uncommon to find a record without a neighbor close by.

Several panoramic range sliders in concert provide the ability to create complex bounding volumes. In that case, we were able to create several bounding solids permitting us to perform a variety of queries based on three dimensions alone. Complementary solids can also be formed, since the range slider permits inversion.

In the future, we intend to perform a detailed user study focusing on user interaction with panoramic range sliders. In addition to comparing the modular range slider to its traditional variant, we will compare it to other continuous, interval-based input controls. Finally, we intend to investigate methods of further improving the panoramic range slider by including widgets such as context menus to provide exact input or lock the behavior of the panoramic range slider.

Acknowledgements

Research reported in this publication was supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Numbers P20GM103424 and 8P20GM103433. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

REFERENCES

[1] R. Spence, Information Visualization: Design for Interaction, 2nd ed. Prentice Hall, 2007.

[2] B. Shneiderman, “Dynamic queries for visual information seeking,” IEEE Softw., vol. 11, no. 6, Nov. 1994, pp. 70–77, [retrieved: Jan., 2014]. [Online]. Available: http://dx.doi.org/10.1109/52.329404

[3] C. Williamson and B. Shneiderman, “The dynamic homefinder: evaluating dynamic queries in a real-estate information exploration system,” in Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, ser. SIGIR ’92. New York, NY, USA: ACM, 1992, pp. 338–346. [Online]. Available: http://doi.acm.org/10.1145/133160.133216

[4] M. Kerr and G. Churchill, “Experimental design for gene expression microarrays,” Biostatistics, vol. 2, 2001, pp. 183-201.

[5] “ACT National and State Scores,” 2011. [Online]. Available: http://www.act.org/newsroom/data/

[6] Digia, Plc., “Qt project,” [retrieved: Nov., 2013]. [Online]. Available: http://qt-project.org

[7] B. Spitzak et al., “Fast Light Toolkit,” [retrieved: Nov., 2013]. [Online]. Available: http://www.fltk.org/

[8] Silicon Graphics Inc., “IRIX 6.5 documentation,” 1998, man page: `SgThumbWheel(3)`.

[9] Kitware, Inc., “KWWidgets,” [retrieved: Nov., 2013]. [Online]. Available: http://www.kwwidgets.org/

[10] M. Osada, H. Liao, and B. Shneiderman, “Alphaslider: Searching textual lists with sliders,” in Proceedings the 9th Annual Japanese Conference on Human Interface. ACM Press, 1993, pp. 365–371.

[11] Image-Line Software, “Fl studio 11,” [retrieved: Jan., 2014]. [Online]. Available: http://www.flstudio.com

[12] Propellerheads AB, “Reason,” [retrieved: Jan., 2014]. [Online]. Available: http://www.propellerheads.se/products/reason

[13] C. Cannam et al., “Rosegarden: music software for Linux,” [retrieved: Jan., 2014]. [Online]. Available: http://www.rosegardenmusic.com

[14] R. Dachselt, M. Frisch, and M. Weiland, “Facetzoom: a continuous multi-scale widget for navigating hierarchical metadata,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI ’08. New York, NY, USA: ACM, 2008, pp. 1353–1356. [Online]. Available: http://doi.acm.org/10.1145/1357054.1357265

[15] R. A. Becker, S. G. Eick, E. O. Miller, and A. R. Wilks, “Dynamic graphical analysis of network data,” in ISI Conference Proceedings, 1989, pp. 119–120.

[16] S. G. Eick, J. L. Steffen, and E. E. Sumner, Jr., “Seesoft-a tool for visualizing line oriented software statistics,” IEEE Trans. Softw. Eng., vol. 18, no. 11, Nov. 1992, pp. 957–968, [retrieved: Jan., 2014]. [Online]. Available: http://dx.doi.org/10.1109/32.177365

[17] C. Ahlberg and B. Shneiderman, “Visual information seeking: tight coupling of dynamic query filters with starfield displays,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI ’94. New York, NY, USA: ACM, 1994, pp. 313–317, [retrieved: Jan., 2014]. [Online]. Available: http://doi.acm.org/10.1145/191666.191775

[18] C. Ahlberg, “Spotfire: an information exploration environment,” SIGMOD Rec., vol. 25, no. 4, Dec. 1996, pp. 25–29, [retrieved: Jan., 2014]. [Online]. Available: http://doi.acm.org/10.1145/245882.245893

[19] Stamen Design, “Oakland Crimespotting update: the pie of time,” [retrieved: Nov., 2013]. [Online]. Available: http://content.stamen.com/node/171

[20] T. Nagel, “A radial range slider,” [retrieved: Nov., 2013]. [Online]. Available: http://www.tillnagel.com/2011/02/radialrangeslider

[21] M. Cho, B. Kim, D. K. Jeong, Y.-G. Shin, and J. Seo, “Dynamic query interface for spatial proximity query with degree-of-interest varied by distance to query point,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI ’10. New York, NY, USA: ACM, 2010, pp. 693–702, [retrieved: Jan., 2014]. [Online]. Available: http://doi.acm.org/10.1145/1753326.1753428

[22] H. Mueller, K. Zatloukal, M. Streit, and D. Schmalstieg, “Interactive exploration of medical data sets,” in Proceedings of the 5th International Conference on Information Visualization in Medical and Biomedical Informatics, 2008, pp. 29–35.

[23] Q. Li, X. Bao, C. Song, J. Zhang, and C. North, “Dynamic query sliders vs. brushing histograms,” in CHI ’03 Extended Abstracts on Human Factors in Computing Systems, ser. CHI EA ’03. New York, NY, USA: ACM, 2003, pp. 834–835, [retrieved: Jan., 2014]. [Online]. Available: http://doi.acm.org/10.1145/765891.766020

[24] J. Heer and B. Shneiderman, “Interactive dynamics for visual analysis,” ACM Queue, vol. 10, no. 30, 2012, pp. 30–55.

[25] User Experience Interaction Guidelines for Windows 7 and Windows Vista. Microsoft Corp., Sept. 2010.

[26] OS X Human Interface Guidelines. Apple Inc., Oct. 2013, [retrieved: Jan., 2014]. [Online]. Available: https://developer.gnome.org/hig-book/3.0/

[27] GNOME Human Interface Guidelines 2.2.2. The GNOME Project, 2013, [retrieved: Jan., 2014]. [Online]. Available: https://developer.gnome.org/hig-book/3.0/