The Effectiveness of Interactive Visualization Techniques for Time Navigation of Dynamic Graphs on Large Displays

Alexandra Lee, Daniel Archambault, and Miguel A. Nacenta

Abstract— Dynamic networks can be challenging to analyze visually, especially if they span a large time range during which new nodes and edges can appear and disappear. Although it is straightforward to provide interfaces for visualization that represent multiple states of the network (i.e., multiple timeslices) either simultaneously (e.g., through small multiples) or interactively (e.g., through interactive animation), these interfaces might not support tasks in which disjoint timeslices need to be compared. Since these tasks are key for understanding the dynamic aspects of the network, understanding which interactive visualizations best support these tasks is important. We present the results of a series of laboratory experiments comparing two traditional approaches (small multiples and interactive animation), with a more recent approach based on interactive timeslicing. The tasks were performed on a large display through a touch interface. Participants completed 24 trials of three tasks with all techniques. The results show that interactive timeslicing brings benefit when comparing distant points in time, but less benefits when analyzing contiguous intervals of time.

Index Terms—Dynamic networks, Information visualization, Large displays

1 Introduction

Dynamic networks are networks that change over time. Nodes and links might appear or disappear at different points in time and attribute values may change. Dynamic networks appear in many domains including social science [22], transportation [30], digital communications [31], epidemiology [45], and others. These networks are difficult to analyze and interpret and can therefore benefit from having interactive visualization techniques applied to them.

Dynamic networks are most commonly visualized by two approaches [6, 12, 41, 46]. One approach is an interactive animated representation where the user can control which moment in time is being displayed. The other is to split the time domain into a series of timeslices and represent them separately as small multiples. This latter approach is currently the most popular in the literature. Multiple studies have shown that the small multiples approach is faster than interactive animation with no significant differences in error rate [4, 5, 26].

The above approaches and experiments all assume uniform slicing at a given level of granularity. However, what uniform duration of timeslice should be chosen? If the timeslices are too coarse, the representation collapses too many events onto the same timeslice, hiding the subtlety and the true order of events within each timeslice. If the timeslices are too fine, there are too many points in time to navigate in the data and analysts will have a hard time remembering timeslices that are off screen when identifying patterns. Interactive timeslicing addresses this issue by allowing the analyst to interactively select the width and location of timeslices with the possibility of representing several of these timeslices at once [44]. Similar methods have been used in other areas [49], but the effectiveness for interactive timeslicing for dynamic graphs is still unknown.

We designed a series of experiments to test an interactive timeslicing approach against interactive animation and small multiples for navigating time in dynamic graphs. We tested these approaches on a touch-based 84" display with 4K resolution that we consider representative of advanced visualization set-ups in current collaborative professional settings [44]. We discuss generalizability implications in Section 10. Currently, there is no evidence that an interactive timeslicing approach will be better; the additional complexity of interaction might be too costly, in terms of time, which could negate all benefits.

We conducted three experiments [1] where participants interactively navigated time to find: a) changes in graph structure at points in time (specific timeslices), b) changes in graph structure across an interval of time (a series of consecutive timeslices), and c) changes in attributes at points in time. For each experiment we had two conditions: near, where the moments in time of interest were close to each other in time, and far, where the moments were further apart in time.

Our results show that interactive timeslicing shortens completion time (small multiples takes 1.42 times longer) and improves correctness (5 percentage point improvement over small multiples) for comparing graph structure at discrete time points. The differences increase almost by a factor of two for temporally distant network events. For finding attribute changes within discrete time points, interactive timeslicing improves completion times (small multiples takes 1.19 times longer for near time intervals). However, for finding graph structure changes over time intervals, small multiples outperforms interactive timeslicing (interactive timeslicing takes 1.42 times longer). These results show how the addition of a relatively simple interactive feature can greatly facilitate the challenging and important analysis tasks of dynamic networks. Our results also further generalize previous findings from the literature on additional tasks that found small multiples has faster completion times when compared to interactive animations for graph analysis with no difference in correctness.

2 Related Work

We review related work for dynamic network visualization techniques along with relevant empirical evaluations. We also provide an overview of network visualization techniques on large displays.

2.1 Dynamic Network Visualization Techniques

Beck et al. [12] separate dynamic graph visualizations into two categories based on their method for encoding time. Time-to-time mappings represent time naturally via the temporal dimension, the most common example of this being animation. Time-to-space mappings use one or more spatial dimensions to encode the temporal information, a common example of this is small multiples.

• Alexandra Lee is with the Dept. of Computer Science, Swansea University and Swansea University Medical School, UK. E-mail: a.s.lee@swansea.ac.uk
• Daniel Archambault is with the Dept. of Computer Science, Swansea University, UK. E-mail: d.w.archambault@swansea.ac.uk
• Miguel A. Nacenta is with the Dept. of Computer Science, University of Victoria, Canada. E-mail: nacenta@uvic.ca

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

[1] All experimental material is available on the OSF at `https://osf.io/bdpnr/?view_only=8d2e29693b714b7d8bb4abd407ad8e56`.

Time-to-Time Mapping Animation was one of the first, and is still one of the most common, approaches for visualizing dynamic data. In interactive animation, an interactive video plays a movie of the evolving graph. The result is reasonably intuitive for node-link visualizations of data (e.g., [8,27,28]). However it requires the user to heavily rely on his or her memory as multiple time points are not visible concurrently. Thus, frequent backwards and forwards navigation through the data is required, incurring interaction costs [5] and additional errors due to fatigue and reliance on memory.

Time-to-Space Mapping Timeline-based visualizations encode the temporal dimension as one or more spatial dimensions. These visualizations can be broken down into four categories: node-link based approaches [23,54], matrix based approaches [14,19,37,66], hybrid approaches [36], and comic-style approaches [7]. Additionally, scalable line charts can effectively show the variations in an attribute value over time [63]. Timeline-based visualizations have the advantage of displaying multiple timeslices on screen simultaneously. However, when event order within a timeslice is essential it is hard to draw conclusions about these events [55]. Also, if the timeslices are separated by large spans of time, interaction is required to scroll back and forth between them for comparison when reordering the timeslices is not possible.

2.2 Experimental Evaluations and Dynamic Data

Much existing work has evaluated the performance of different time and visualization types within a mapping.

Saraiya et al. [53] compared time-to-time and time-to-space mappings for node-link diagrams when interacting with multidimensional data. Animation performed better for two points in time, but tasks involving more time steps were better served by timeline-based approaches. A number of studies have demonstrated advantages for time-to-space mappings on dynamic graphs [4,5,26] and visualization in general [62]. These studies use linear interpolation (fading nodes in/out from the visualization) and find that small multiples is faster with no significant differences in terms of error rate. Boyandin et al. [17] find that animation facilitates findings on adjacent time steps but that small multiples allow the discovery of patterns which last for greater periods of time. Hybrid approaches mixing animation and timelines can, under certain conditions, produce better results than animation or timeline approaches alone [52].

The majority of these experiments have focused on the structural properties of the network first and the time navigation second. Also, all the above experiments assumed a uniform timeslicing selected beforehand whereas many recent techniques for graph visualization do not make this assumption [44,57,58,64]. For dynamic graphs that are long in time (for example, events lasting seconds over months of data), no experiments have been run. Also, interactive timeslicing has not been evaluated. We present three experiments that evaluate user interaction with the time dimension for long in time dynamic graphs on a large touch display.

2.3 Visualization on Large-Screen Devices

Visualization on large displays has a long history and is appealing for a variety of reasons. In the past, larger displays (usually composites of many smaller displays [9,15,34,48,59]) were, due to technology constraints, the easiest way to increase the available pixel count. This, in turn, increased the ability to display detail or more data items. Increased number of pixels and larger size has been shown to increase performance and has perceived benefits [25,60,61].

Although modern display technology has reached pixel densities that make the argument about pixel counts largely irrelevant (a state-of-the-art 15” display can have as many pixels as an 88” display from just a few years ago), there are still reasons why large displays are desirable for visualization. First, they enable larger numbers of people to work on the same data [39]. Second, people interacting with large displays seem to benefit from physical navigation [2,10,40], which might improve memory and performance [39]. The ability to change the distance to the display easily and naturally by stepping back and forth also enables a natural zoom experience and supports visualization techniques that would be hard or awkward with personal displays (e.g., [38,40,47]). Incidentally, a larger display will also reduce problems with the precision of touch interaction (e.g., the fat finger problem [56]) simply because the content will normally be larger.

Perhaps due to these reasons, research investigating and translating visualization to large displays is still active (e.g., [43,44,50]). From the information available in literature and our own experience, we speculate that large displays are likely to offer the best environment for the visualization of complex and high-density information such as dynamic networks, particularly when scalability in the time dimension is required. The specialized nature of this kind of analysis also means that the extra cost of procuring large displays is usually well justified, and many research environments already offer medium to very large displays. Hence, we carried out our implementation on a relatively large display that matches the use of dynamic network visualizations that we envision in the near future. Our implementations do not include any interactive or visual features that prevent use with a small personal display. Therefore, we do not expect that the comparisons between human performance or preference between the techniques would vary in a smaller display, although this will need to be supported empirically in the future.

3 Experiment Interfaces

For this experiment, we consider three dynamic graph visualization techniques: small multiples, interactive animation, and interactive timeslicing. Animation and small multiples were chosen because they are widely considered the dominant alternatives in dynamic graph visualization [12], are in common use, and have been revisited in recent data visualization experiments for small displays [21]. Interactive timeslicing represents a novel alternative that is promising but has not yet been compared with the other two approaches [44].

Node-link diagrams were used for all graph representations as they are popular in media and are well explored in literature. All approaches were implemented to work on a large touch-enabled display (see Apparatus).

3.1 Interactive Animation

Interactive animations present the dynamic graph as an interactive film, with the user being given control of playing the dynamic data via a slider that can travel both forwards and backwards in time. Nodes and edges fade in and out of the drawing area as they are inserted or removed from the network.

For this study our animation interface has four components. The component labeled (A) (see Figure 1, interactive animation) was the main timeline which showed the number of edges within the dataset aggregated at the hour level. The component labeled (B) was an interactive time window selection. Users would touch and drag to select a time window from the longer time series, with component (B.1) also selecting the 6 hour time period immediately preceding the user selected time window. An alternative solution would have been to animate directly on the long timeline. However, this could potentially introduce a confound into our experiment as the animation would need to consider a much larger amount of data (the entire long-in-time dynamic graph) rather than a shorter animation around a given time window. Therefore, we decided to allow the user to select the animation window on the longer timeline around the target area.

Component (C) is the flattened graph representation of the time window selected by component (B.1). This static graph allows the participant to enter an answer without navigating through the network. Component (D) is the interactive animation window — when the time range was first selected, component (D) showed a flattened, static, representation of the graph within the selected time range. The participant pressed and dragged on area (D.1) to control the rate and position of the animation. A purple line within (D.1) follows the participant’s finger to show the current temporal graph position relative to the timeline.

3.2 Small Multiples

Small multiples uses many interrelated graphs to represent time. It is analogous to a comic book representation for time where the evolution

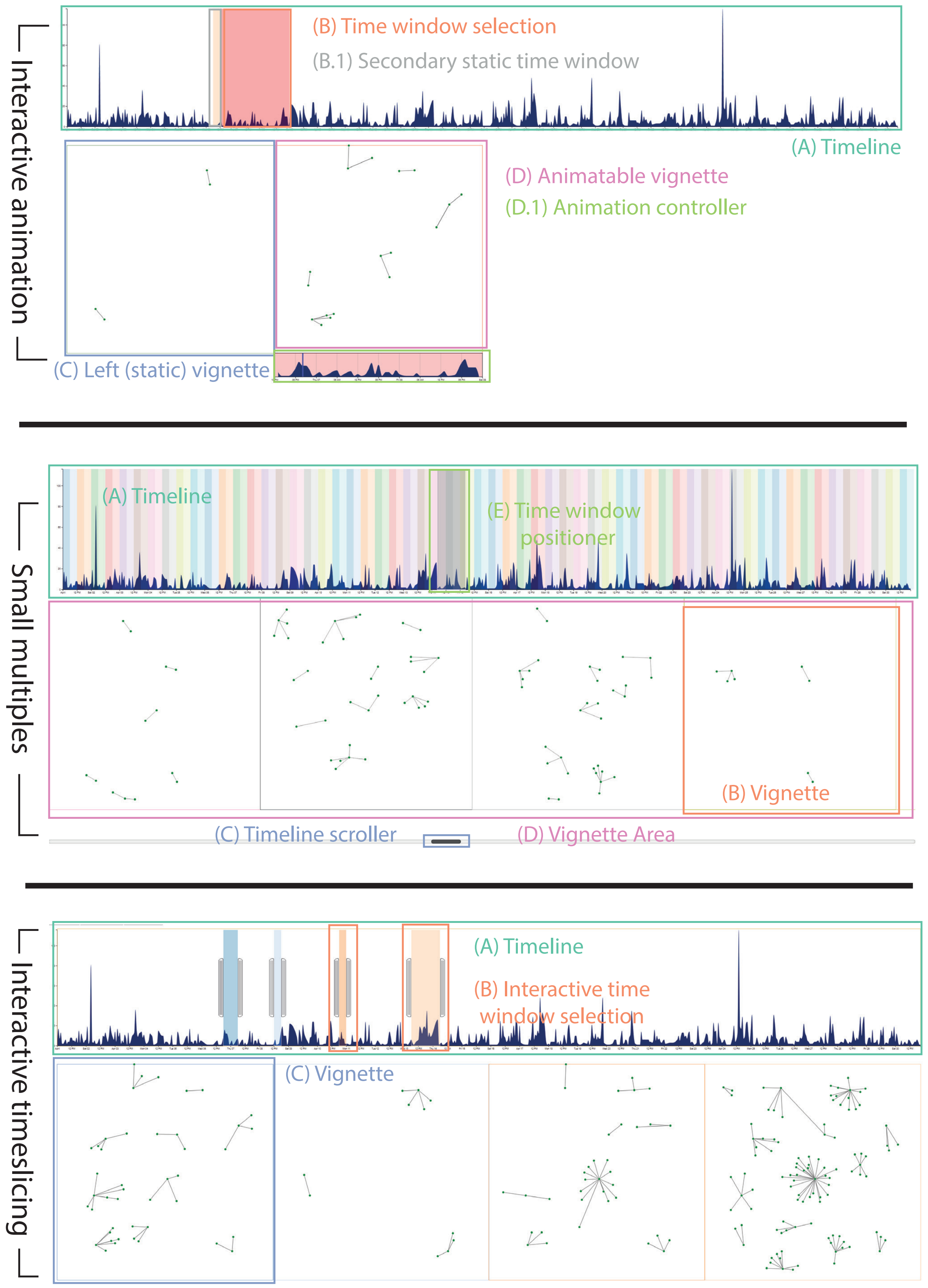

Fig. 1. The interfaces used in this experiment. Interactive animation is at the top, small multiples in the middle, and interactive timeslicing at the bottom.

of the data can be tracked by reading across the frames from left to right. We use the term small multiples as it is most similar to those in other experiments [4, 5, 21, 26, 51]. For temporal networks, it is common to uniformly timeslice data by flattening each timeslice into its own window, with these graphs then drawn next to each other in a timeline pattern. Small multiples can be used for most forms of dynamic data; in dynamic graph visualization, it can be used both with matrices and node-link diagrams.

For the small multiples representation in these experiments, our study dataset (Section 5.3) was divided into blocks of 6 hours, giving a total of 120 timeslices which were translated into vignettes. Along the top of the screen, labeled (A), is the timeline. Each block of color defines one 6 hour time range, and each block of color also corresponds to the border color of the relevant vignette shown in (B). The vignette area was designed to only show four small multiple representations simultaneously due to screen size and readability constraints. The remaining vignettes could be accessed by moving the scroll bar at the bottom of the screen, (C), touching and dragging on the vignette area, (D), or by moving the light gray time-positioner on the timeline, (E). The gray time-positioner, (E), helped participants map the current position of the vignette area to its temporal position within the entire time series without performing color comparisons of the vignette borders.

3.3 Interactive Timeslicing

This interface was based on a previous interactive timeslicing design [44] where users could define custom time ranges of any desired size or order and have the individual representations positioned based upon the order of user interactions.

Component (A) is the timeline that is common to all of our experimental interfaces. Participants touched and dragged at any position on the timeline to create a new time window selection of any size (B). A time window selection created a new vignette (C) containing a flattened, static, representation of the graph. A time window selection could be any multiple of 6 hours, with the edges of the selection snapping to the same timeslice intervals as all conditions — this was to avoid small off-by-one errors that may seriously hinder the ability of a participant to successfully complete any given task.

The maximum number of interactive timeslices that could appear onscreen at any time was four. This limitation was introduced to avoid giving interactive timeslicing an artificial advantage over small multiples. Interactive selection of timeslices in this way allows the participant to make independent time windows at very distant points in time for the network. However, there is an interaction cost associated with creating the time windows that is higher than both animation and small multiples: the location in time needs to be determined first and then the participant needs to draw a much more precise window both in terms of its width and position.

4 Experiment and Research Questions

Our general goal is to provide empirical evidence that can support the design of better interfaces for time navigation when exploring dynamic networks. Thus, we designed three experiments, each testing a different task. For each experiment we seek to answer the following questions:

- Q1 Which interface will have the lowest completion times for the selected tasks?
- Q2 Which interface will be the most accurate for completing the selected tasks?
- Q3 How does distance in time between data elements of interest affect the performance with the different interfaces?

5 Experimental Approach

We designed a series of three experiments through the following iterative process. First, we looked at existing relevant taxonomies of tasks [1, 3, 42] and at findings from previous observations of analysts working with dynamic networks [44]. Then we filtered out tasks that do not involve navigation between at least two given time points. We excluded tasks that only involve single time points, since are we primarily interested in challenges specific to dynamic, rather than static, networks. From the remaining tasks we selected three tasks with the following criteria: a) we preferred low-level tasks that might be components of larger tasks, and b) we preferred tasks that were very different from each other and where the focus was on different elements of the data (e.g., the structure of the network or the variation of attributes) and how they varied over time.

The chosen tasks can be broadly summarized as:

- E1 Detecting graph structure changes at discrete points in time.
- E2 Detecting graph structure changes over a time interval.
- E3 Detecting single attribute change at discrete points in time.

We run the three experiments with the same participants in the same session.We prioritize comparing results across interfaces on the same task; to reduce noise from order effects on our comparisons of interest, we run the three experiments in the same order for all participants (i.e., we do not randomize experiment order). Experiments took place in early March 2020 and took approximately 85 minutes to complete.

5.1 Participants

Twenty-four unpaid volunteers (8 identified as females, and 16 as males, age range of 18 to 32) from our department participated in the study. The number of participants was decided in advance using our prior experience from the four pilots and to ensure proper counterbalancing. We used a simple questionnaire to screen participants without typical color vision and those who did not have basic computer and mathematics experience. We did not require prior knowledge of networks.

Before beginning the first experiment, participants filled a short demographic questionnaire indicating their familiarity with networks and network graph visualization using a 1–5 Likert scale. Four participants gave their familiarity as 1 (no knowledge), six as 2, nine as 3, three as 4, and two as 5 (high knowledge).

5.2 Apparatus

The interfaces were built in JavaScript, primarily leveraging the D3.js library [16] with some modifications. Conditions ran in a Chromium browser window which was wrapped in a QT front end interface, with a Python Flask back-end. Implementations of the interfaces are identical across the three experiments; visual indicators for the task and methods to indicate the answer vary with the task and are described within each experiment.

All three of these approaches were implemented for an 86” wall display with 4k resolution (3840 by 2160 pixels) mounted at 90cm from the floor.

5.3 Experimental Dataset and Graph Layout

The dataset used in these experiments is a filtered version of a previously collected dataset [22] which describes interactions between anonymized actors on Instagram who liked or commented on non-suicidal self-injury posts. Edges have unique identifiers, have a specific occurrence time (down to second precision), and do not have a duration (i.e., are considered atomic, only measured at the time of posting). A duration of one hour is assigned to each edge at its posting time to ensure it is visible and lies completely within the six hour time window. Additional attributes of the dataset were removed for anonymity; hence the dataset contains only nodes, edges, and the time that the nodes and edges appeared in the dataset.

The dataset is representative of typical dynamic networks but was substantially filtered to reduce its size to make our tasks feasible. We removed single interactions between actors (cases where two actors only interacted with each other once), self-interactions, actors that received less than 250 interactions, and actors who only interacted with removed actors. This allowed a stable graph layout and made task duration appropriate to the available time. The final dataset consists of 776 distinct nodes and 8182 edges over a 30-day period of time. It had an approximate average density of 11 events per hour, or 66 events per 6 hour timeslice. As this dataset is based on real data, the event distribution was non-uniform and therefore some time periods had a higher event density, while others had a lower event density. This variability over long time periods allowed for very different graphs to be presented at the time points and intervals in our tasks, which increases the generalizability of trials.

Our dataset is long in time with a fine grained temporal resolution (hour resolution over a month of data), meaning that there are hundreds of timeslices. Typical timeslice-based approaches [18], in particular linking strategies, generally do not scale to hundreds of timeslices as nodes move unnecessarily with the many inter-timeslice edges being artificially inserted [57]. As we wanted to control for graph layout across conditions we could not draw the graph on demand as the user navigated through time. Thus, we needed to draw the full dynamic graph beforehand. Event-based dynamic network drawing techniques could provide a solution [57, 58] but they only scale to about 5000 events. Therefore, we computed a fixed layout for nodes and edges that uses pinning strategies so that nodes retain position whenever they appear, allowing the drawing stability to consistently support the mental map across conditions for all experiments. The final graph layout was generated first with the Visone aggregation strategy [20] (all events collapsed down to a single timeslice) and then cleaned slightly in Gephi [11], using its implementation of Fruchterman and Reingold's algorithm [29]. Although more scalable algorithms are available [35], our graphs are relatively small in terms of the number of nodes and edges. This allows us to use standard force-directed approaches instead of multilevel ones. The resulting layout still contained some overlapping nodes and edges; we manually adjusted overlapping items.

5.4 Experimental Design and Procedure

All three experiments share an identical structure in terms of factors, conditions, and repetitions. The main manipulation of interest is *interface*, with the three interfaces `Anim`, `Mult`, and `Int TS` as levels—see Section 3. One factor is temporal distance (near and far), which manipulates the separation of the target timeslices or the length of the time interval involved in the task. In the near condition, timeslices had a separation of 18 hours (three timeslices) while in the far condition timeslices had a separation of four days (16 timeslices).

We performed a within-participant design with participants completing 3 repetitions for each cell, for a total of 18 trials, as well as two additional easy tasks on each interface for training which was discarded before data analysis. Participants carried out trials (including training) in three blocks of 8, with 15 second breaks between each block. Possible ordering effects were counterbalanced using Latin squares for each participant. A participant always did the interfaces in the same order for each experiment that they completed. Trials in the near condition always took place before trials in the far condition. We did not counterbalance temporal distance because we are not interested in the quantitative comparison of performance between near and far conditions.

Prior to the real trials of each interface condition participants received a tutorial on the specific interface and carried out one trial example. Upon completion of each experiment participants ranked each condition on a scale of 1 (best) to 3 (worst), according to their preferred interface for completing the type of tasks tested. The experimenter also collected qualitative notes during the experiment. Participants had the opportunity to make other comments and share their thoughts about the tasks, interfaces, and hardware.

5.5 Statistical Methodology

The statistical methodology was decided during the experimental design process and recorded before data collection. All three experiments employed the same methodology.

Correctness and completion time were measured and analyzed separately for all three experiments. Completion time was measured as the number of seconds (s) to complete each task for all three experiments. Measurements for correctness varied for each experiment, and these measures are detailed in their respective sections. However, for all three experiments, correctness was measured on a $[0, 1]$ interval (with 1 corresponding to 100% correctness).

As we were interested in determining the performance of the three interfaces under near and far conditions, we chose to divide the data of each experiment by the near and far factor before beginning the analysis. The completion times and correctness of the three repetitions in each cell was averaged per participant. Completion time was log transformed (log2) before analysis and compared through pairwise, two tailed t-tests. We were uncertain if the distribution of correctness data would follow a normal distribution for our measurements, as a result we applied a Shapiro-Wilk test with $\alpha = 0.05$ to each condition of near and far for each experiment separately. For all three experiments, the correctness data did not usually follow a normal distribution. Therefore, two tailed, pairwise Wilcoxon signed rank tests were used for correctness in all three experiments. For each experiment, six pairwise comparisons (three for near and three for far) for time and six pairwise comparisons for correctness were performed. Holm–Bonferroni corrections were used to determine significant results.

The results are presented in the next section. In all result figures, blue corresponds to animation (`Anim`), red to small multiples (`Mult`), and orange for interactive timeslicing (`Int TS`). The mean is indicated using a circle and the median using a square. Error bars represent bootstrapped 95% confidence intervals computed using 250,000 repetitions. We report p-values to three decimal places in all figures [13, 32, 33, 65]. Solid lines indicate $p < 0.05$ and dashed lines $p \geq 0.05$.

6 Exp. 1: Graph Structure Changes at Points in Time

This experiment tests completion time and accuracy for tasks where the structure of the graph had to be compared at two separate time points. We suspected that interactive timeslicing would perform the best for this task (**Q1** and **Q2**). In addition, we anticipated that interactive timeslicing would perform much better than animation and small multiples for the far condition. However, we were less sure of the performance of the remaining two interfaces on this condition of the experiment as they have not been tested and were not designed for exploring distant points in time (**Q3**).

Task and Procedure. The task prompt provided to participants for experiment one was 'each pair of red (start) and blue (stop) lines signify a timeslice. In the first timeslice, click on all edges that disappear at least once in all other timeslices.' The target timeslices of six hours each began with a red line and finished with a blue line. Participants completed the task by selecting edges in the first window that did not appear in the other window.

The video figure in the supplementary material illustrates how participants answered this task on all three interfaces. For all interfaces, answers to the question were entered through lasso selection on the leftmost vignette to control for answer entry. Edges selected using the lasso tool were coloured red and increased $4\times$ in width and could be de-selected.

Animation. Participants dragged on the timeline starting from the end of the first red-blue pair of timeslice demarcation lines, ensuring that the vignette to define an answer was correctly created. This action would generate two vignettes: a left vignette displaying the selected time interval of the first pair of timeslice lines as a static, flattened, graph for answer entry; and a right vignette which initially displayed the flattened, static, graph covering the whole time range selected by the participant during the original time selection operation.

A timeline appeared at the bottom of the right hand vignette displaying a zoomed version of the selected area of the main timeline, with task target timeslice areas shaded in blue. Participants were able to touch and drag on the timeline to animate the right hand vignette. Touching and dragging in the left vignette would activate the lasso tool to allow participants to select, or deselect, edges for their task answer.

Small Multiples. Participants used either the grey time positioner (Figure 1, small multiples, item E) or the scroll bar attached to the vignette area (Figure 1, small multiples, item C) to navigate to the correct position in time. Task relevant vignettes were distinguished from non-task vignettes by increasing the border thickness $4\times$. Participants were able to touch and drag in the left vignette to use the lasso tool to define the answer set.

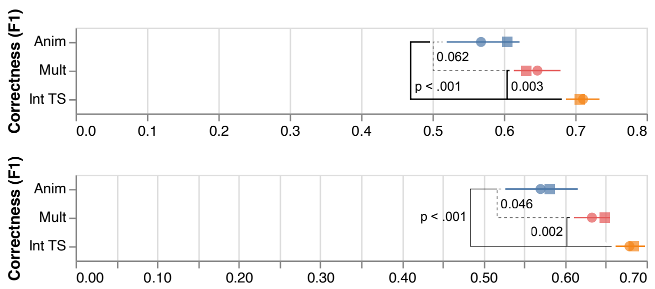

Fig. 2. Experiment 1 correctness for the near (top) and far (bottom) conditions as computed by the F_1 score (Equation (1)). The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

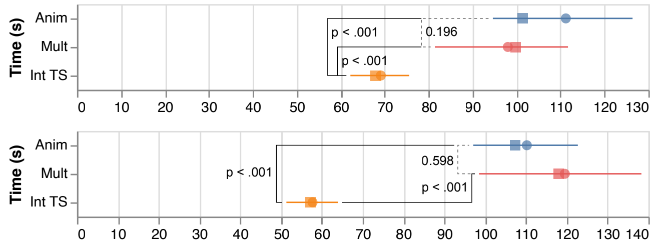

Fig. 3. Experiment 1 completion time (s) in seconds for the near (top) and far (bottom) conditions without log transform. The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

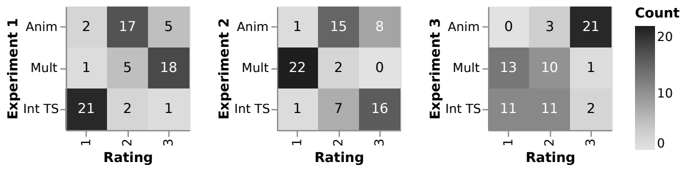

Fig. 4. Participant interface rankings. Cells annotated by number of votes and darker cells correspond to more votes. Rating is 1st, 2nd, or 3rd.

Interactive Timeslicing. Participants touched and dragged on the correct area of the timeline to create a time window selection (Figure 1, interactive timeslicing, item B). To avoid small selection errors these user-created time windows always readjusted to the nearest 6 hour boundary, ensuring that the selection was the exact time period required by the task. After the creation of a time window a corresponding vignette appeared below the timeline (Figure 1, interactive timeslicing, item C). Touching and dragging in the left-most vignette with the lasso tool defined an answer to the given task.

Measurements. Time (number of seconds) and accuracy were measured for this experiment. Accuracy involved comparing the set of edges selected by the user and the set of edges present in the correct answer of the question. In order to evaluate correctness in this case, we employ a method from pattern recognition and information retrieval. A perfect answer would have perfect *precision* (p) (no edges outside the correct answer are selected) and perfect *recall* (r) (all of the edges in the correct answer are selected). Precision and recall can be combined together into the F_1 score to give a measure between $[0,1]$:

$$F_1 = 2\frac{pr}{p+r} \tag{1}$$

The value of p and r are defined in the following way. Consider two edge sets: the set of participant answer edges (X), and the set of correct answer edges (Y).

$$p = \frac{|X \cap Y|}{|X|} \tag{2}$$

$$r = \frac{|X \cap Y|}{|Y|} \tag{3}$$

Results. Correctness and completion time are shown in Figures 2 and 3 respectively. After a Holm–Bonferroni correction, we found significant differences in terms of correctness for the near condition with interactive timeslicing outperforming animation with a difference of 14 percentage points ($W = 82$, $p < 0.001$) and small multiples with a difference of 6 percentage points ($W = 148$, $p = 0.003$). On the far condition, interactive timeslicing outperformed animation with a difference of 11 percentage points ($W = 109$, $p < 0.001$) and small multiples with a difference of 5 percentage points ($W = 143$, $p = 0.002$). No other differences were statistically significant. In terms of completion time, we have the same pattern. On near, interactive timeslicing outperforms animation with a difference of $42.1s$ (animation $1.61\times$ slower) ($t = 5.76$, $df = 23$, $p < 0.001$) and small multiples with a difference of $28.9s$ (small multiples $1.42\times$ slower) ($t = 4.28$, $df = 23$, $p < 0.001$). On far, interactive timeslicing outperforms animation with a difference of $52.5s$ (animation $1.91\times$ slower) ($t = 8.04$, $df = 23$, $p < 0.001$) and small multiples with a difference of $61.8s$ (small multiples $2.07\times$ slower) ($t = 8.01$, $df = 23$, $p < 0.001$).

Participant Survey. After completing the experiment, participants ranked each condition on a scale of 1 (best) to 3 (worst) to indicate their preference for conditions to complete the given task type. These ranking responses are in Fig. 4. For this experiment 87.5% of participants indicated that they preferred the interactive timeslicing, with interactive animation second, and small multiples as the least preferred option. The primary criticism of small multiples was that, for tasks involving the far in time condition, it was impossible to put both timeslices on screen simultaneously in order to directly compare them. Participants found it challenging to remember edge positions and edge presence when they had to scroll through many intermediate representations to reach the comparison target.

The memory cost was also probably a factor in the ranking of animation. However, this was less pronounced as the interactive animation interface made it simpler to switch backwards and forwards in time without having to view intervening timeslices.

Summary and Discussion. For both the near and far conditions, interactive timeslicing outperforms animation and small multiples in terms of correctness and completion time, confirming our conjectures for all research questions. There were no other significant differences found in the experiment. Interactive timeslicing was primarily designed for dynamic graphs spanning a long interval of time and specifically for the case of comparing distant points in time, as required by the task. Animation incurs higher interaction costs by requiring interaction to play the animation back and forth between the distant time periods. Small multiples requires the participant to scroll back and forth between the distant time periods. In addition, both animation and small multiples require participants to remember the structure at distant time points, whereas interactive timeslicing is able to show representations simultaneously and side-by-side on screen. It seems that the added interaction cost of interaction timeslicing is offset by this benefit. It is important to note that we did not see a significant difference between animation and small multiples for either near or far. This could be due to the fact that neither of these interfaces were designed with long time series in mind. Thus, on the task of comparing distant points in time, these interfaces were too taxing on memory and interaction, which dominated the result.

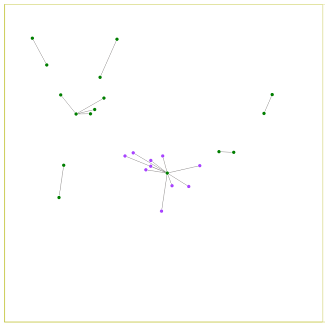
Fig. 5. An example of a vignette with a cluster of purple nodes, as shown to participants during Exp. 2

7 Exp. 2: Graph Structure Changes Over a Time Interval

This experiment tests how interactive timeslicing performs over a time interval when examining a change in graph structure. A time interval is several consecutive timeslices. In terms of our research questions, we conjectured that interactive timeslicing would not perform as well on this task type because it is mainly designed for time points and not continuous intervals (**Q1** and **Q2**). We also felt that the performance of interactive timeslicing would decrease with far trials because a longer interval of time needed to be considered and more information needed to be remembered (**Q3**).

Task and Procedure. The on-screen prompt provided to participants for experiment two was: 'select a timeslice between the red lines where the cluster of purple nodes is most dense.'

The video figure in the supplementary material illustrates how participants answered this task on all three interfaces. Participants saw a pair of red lines on interface timelines (see Figure 1, (A), on all interfaces), indicating the beginning and end of the time interval of interest. Successful completion of the task involved investigating every six hour timeslice within this interval to identify the timeslice in which there were the highest number of connected purple nodes (see Figure 5 for an example).

When participants were confident that they had found the correct answer, they would tap a button above the timeline to switch into 'answer entry' mode. In this mode, participants would slide a green time window (the fixed size of one timeslice) to the position on the timeline containing their answer.

Measurements. Time (s) and accuracy were measured for this experiment. Accuracy involved comparing the number of edges between purple nodes in the selected timeslice (n_s) to the timeslice where the number of edges is a maximum (the correct answer) (n_a). In order to do this, we use the following measure:

$$c = 1 - \frac{n_a - n_s}{n_a} \tag{4}$$

The value of this measure is 1 when the correct answer is selected and diminishes to zero with a window of fewer and fewer edges between the purple nodes.

Results. Correctness and completion time are shown in Figures 6 and 7 respectively. After a Holm–Bonferroni correction, we found no significant differences in correctness between interfaces in neither near nor far conditions. In terms of completion time on near, small multiples outperformed animation by $9.3s$ (animation $1.24\times$ slower) ($t = 3.31$, $df = 23$, $p = 0.003$) and interactive timeslicing outperformed animation by $9.1s$ (animation $1.24\times$ slower) ($t = 3.54$, $df = 23$, $p = 0.002$). On far, small multiples outperformed both animation $17.8s$ (animation $1.34\times$ slower) ($t = 4.01$, $df = 23$, $p < 0.001$) and interactive timeslicing $21.8s$ (interactive timeslicing $1.42\times$ slower) ($t = -6.67$, $df = 23$, $p < 0.001$). No other significant differences were found.

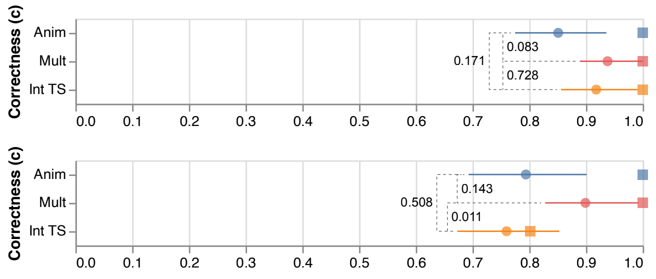

Fig. 6. Experiment 2 correctness (Equation (4)) for the near (top) and far (bottom) conditions. The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

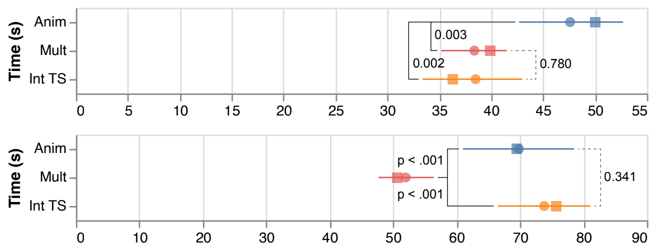

Fig. 7. Experiment 2 completion time in seconds for the near (top) and far (bottom) conditions without log transform. The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

Participant Survey. After completing the experiment, participants ranked each condition on a scale of 1 (best) to 3 (worst) to indicate their preference for conditions to complete the given task type. These ranking responses can be seen in Fig. 4. For this experiment, 92% of participants preferred small multiples for completing tasks of this type. Animation ranked second and 67% of participants ranked interactive timeslicing as the worst.

There was widespread frustration among participants while using the interactive timeslicing condition during this experiment. The comparison of a number of time points within a time interval required much interaction effort. A large amount of very precise interactions were required to position and re-position the timeslices; in contrast, small multiples simply required scrolling the vignette display area while looking at the screen. For this task type interactive timeslicing was also vulnerable to 'fat finger' [56] problems, where participants aimed to carry out one operation but accidentally triggered a different one due to imprecise touching of the screen. A common example occurred when a participant tried to move a time window selector but instead activated a resize operation by selecting a handle for the selected time window. The participant then had to return the time window to its previous size and attempt to carry out the re-positioning operation again.

Summary and Discussion. For this experiment, we found no significant differences in correctness. Therefore, we have no evidence that any interface was more accurate than another, but some of the interfaces were more efficient. When the time interval is smaller, we can conclude that animation is slower than both small multiples and interactive timeslicing. Interactive animation is the only one of these interfaces where all timeslices of the time interval must be remembered in order to compare them. For both small multiples and interactive timeslicing some of the representation of the time interval can be offloaded to the interface as multiple timeslices are shown. When the time interval is wider, we can conclude that small multiples is faster than both animation and interactive timeslicing. As the interval of time considered increases, interactive timeslicing is more strongly affected as this technique is based on individual timeslices. The implementation of small multiples from this experiment provides a more natural interaction with a time

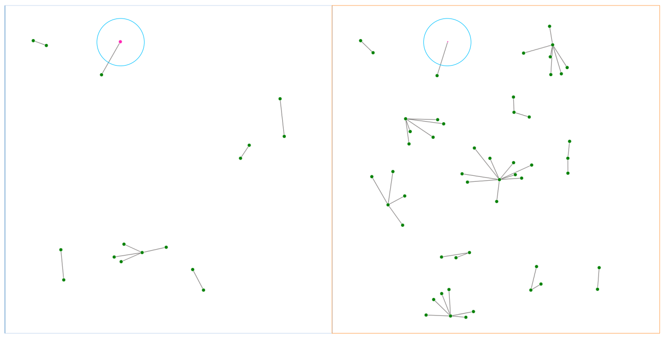

Fig. 8. Two vignettes, as seen during task 3, with a standard size pink node (left) and a small size pink node (right). Pink nodes are highlighted with a blue circle for the purposes of readability, but this was not present on the experiment interfaces.

interval as all timeslices must be contiguous.

8 Exp. 3: Attribute Changes at Points in Time

This experiment tests performance when reading attribute values at multiple, disjoint, time points. In terms of research questions, we thought that interactive timeslicing would perform well here (**Q1** and **Q2**). We felt that this difference would increase for the far level (**Q3**).

Task and Procedure. The instruction for this third experiment was 'each pair of red (start) and blue (stop) lines signify a timeslice where a pink node is present. In which timeslice is the pink node smallest?'. The video figure in the supplementary material illustrates how participants answered this task on all three interfaces. A randomly selected node, present throughout the interval of interest, was given a minimum attribute value at precisely one of those timeslices and a maximum value in all others. Bright pink was chosen as the color of the target node for contrast reasons to minimize visual search time. The standard node size is 3.3× bigger than the size of the smaller answer node. An example of a normal pink node vs. the smallest pink node can be seen in Figure 8.

The procedure for submitting answers for this task was the same as **Exp 2**, with participants using a button to enter 'answer mode' and moving a timeslice, of fixed size, to their answer position.

Measurements. Time (s) and accuracy were measured for this experiment. There was only one correct answer where the attribute was at its minimum value. Therefore, a score of 1 was recorded for each task answered correctly and 0 for an incorrect answer.

Results. Correctness and completion time are shown in Figures 9 and 10 respectively. After a Holm–Bonferroni correction, we found no significant differences between the interface conditions in neither the near nor the far condition. All interfaces had median 100% correctness for this experiment for both near and far conditions. In terms of completion time on near, all pairwise differences were significant with interactive timeslicing outperforming both small multiples by 6.3s (small multiples 1.19× slower) ($t = 4.45$, $df = 23$, $p < 0.001$) and animation by 13.6s (animation 1.42× slower) ($t = 8.52$, $df = 23$, $p < 0.001$), and small multiples outperforming animation by 7.3s (animation 1.19× slower) ($t = 4.80$, $df = 23$, $p < 0.001$). On the far condition, small multiples outperformed animation by 26.1s (animation 1.96× slower) ($t = 12.7$, $df = 23$, $p < 0.001$) and interactive timeslicing outperformed animation by 26.7s (animation 2.00× slower) ($t = 11.8$, $df = 23$, $p < 0.001$). The remaining pairwise difference between interactive timeslicing and small multiples was not significant.

Participant Survey. After completing the experiment, participants ranked the interfaces on a scale of 1 (best) to 3 (worst) in order of preference to complete the given task type. For full results see Fig. 4. There was no clear preference for a single interface for completing this task type. However, animation was strongly disliked with 87.5% participants giving it a rank of 3. There is a relatively low interaction cost for

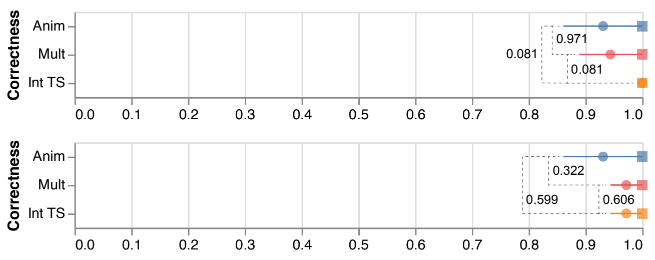

Fig. 9. Experiment 3 correctness for the near (top) and far (bottom) conditions. The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

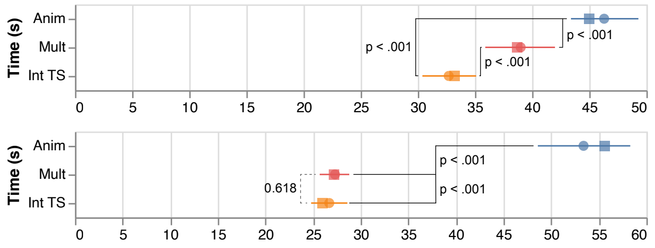

Fig. 10. Experiment 3 completion time in seconds for the near (top) and far (bottom) conditions without log transform. The dot represents the mean, square the median, and 95% CIs shown. Pairwise lines indicate p-values with solid lines indicating significant differences.

small multiples and interactive timeslicing for this task, whereas animation required participants to scroll across the timeslices. Interactive animation also required a much higher level of concentration for this task type than the other interfaces, with participants having to identify the target node and then remember its position while animating the graph. The opposite is the case for interactive timeslicing and small multiples where all nodes and edges were always visible, meaning participants simply had to identify the pink node in each time window of interest rather than attempt to remember the previous node position.

Summary and Discussion. For this experiment, no significant differences were found in terms of correctness. Hence there is no evidence supporting superior accuracy of any of the interfaces. In terms of completion time for near, all pairwise differences were significant with interactive timeslicing the fastest, followed by small multiples, and then animation. As the near condition is closest to previous experiments [4, 26], small multiples outperforming interactive animation is consistent with this result. For far, we did not see a difference between interactive timeslicing and small multiples where there was one for near. However, animation is significantly slower than both interfaces. There could be many reasons why we did not find a difference between small multiples and interactive timeslicing. One possible interpretation is that the task is less demanding on participant memory (remembering a single node instead of a collection of edges), and thus, the difference is less pronounced. Further experimentation is required to test this hypothesis.

9 General Discussion

We first summarize the main results from the experiment based on the research questions from Section 4. Answers to the questions vary depending on the task, with Experiment 1 and 3 showing similar patterns but different from Experiment 2.

In Experiments 1 and 3, task completion times with the different techniques (Q1) show the clear advantage of interactive timeslicing over the other techniques. We take this as indication that interactive timeslicing facilitates time navigation between discrete points, which is what the tasks from Experiments 1 and 3 have in common. There is one exception: in the far condition of Experiment 3, small multiples and interactive timeslicing showed similar completion times (i.e., we observed a difference between the near and far conditions — Q3). We attribute this to a reduced memory requirement for that condition. Occasionally small multiples visibly outperformed animation. In accuracy (Q2), interactive timeslicing also showed a clear advantage over the other two techniques in Experiment 1, but not in Experiment 3, where accuracy was high for all interfaces and statistically indistinguishable.

Experiment 2 has very different results. Small multiples was faster than the other techniques in the far condition (Q1), although statistically indistinguishable from interactive timeslicing in the near condition. Accuracy measurements in this experiment show fairly large differences in means, but large variance in the trials prevents us from finding statistically reliable differences between interfaces in both near and far conditions (Q2). A possible explanation of why small multiples performed significantly better is that it naturally represents consecutive time windows that can be scrolled through easily.

When considering the survey data, no interface was preferred by the majority of participants for all tasks. Instead, interface preference is task dependent. For the first and second experiments the rankings did indicate clear interface preferences for the completion of those task types; interactive timeslicing and small multiples ranked first for those experiments, respectively. In contrast to Experiment 1 and 2, there was no clear preference for a single interface for completing the third experiment. From our own observations of Experiment 3, there seemed to be fewer 'fat finger' [56] interaction mistakes with interactive timeslicing than in Experiment 2. It is likely that this is because there was no requirement to move or resize a time window after the initial definition stage (barring participant error). One participant remarked that their version of a perfect interface would be small multiples with the ability to make selected vignettes appear and disappear to better facilitate side-by-side comparison of highly distant time points.

Interactive animation is often slower and less preferred but with no difference in terms of correctness, meaning it is a less efficient way of finding the correct solution. On Experiment 1, it is significantly slower than interactive timeslicing. On Experiment 2, it is significantly slower than small multiples. On Experiment 3, it is significantly slower than the other two interfaces. Thus, for tasks involving time navigation as tested in these experiments, we confirm some results of other experiments on dynamic data [4, 5, 21, 26, 51]. One possible conjecture for why it is slower is that the participant has no idea where to look in the animation when undertaking an exploratory task. Although there are some preliminary results [51], it remains an open question whether animation works well for explanatory tasks where a presenter can point out regions of interest for another viewer to understand.

Brehmer et al. [21] compared the efficiency of animation and small multiples on mobile phones for animated scatterplots. The result of this previous experiment found that small multiples was usually faster than animation with no difference in correctness. One could view our study as a somewhat analogous test on large displays with a touch interface that confirms several results comparing the interactive animation and small multiples conditions.

10 Limitations and Future Work

As with all experiments, experimental design choices cause limitations in result interpretability. One such choice is that we prioritized counterbalancing the interfaces, and not the experiments or near/far factor, in order to reduce experimental noise. Thus, later experiments might have had more tired participants and earlier experiments had less training. Far tasks might have benefited from the experience of having done the near tasks first. Nevertheless, we hypothesize that it is unlikely that training or fatigue might have affected the interfaces differently.

All trials in our experiment use parts of the same dynamic network data. Although this dynamic graph is long in time with much variability in the different time points and intervals of the different trials, it is important to test other datasets in the future for the sake of generalizability and to further explore which types of structures or values might affect the different interfaces.

In order to keep a reasonable experiment length, we only tested three task types, but understanding how these interfaces perform for a wider set of tasks (e.g., additional ones from [3] or [1, 42]) would produce further insights. We only test our interfaces with node-link diagrams despite matrices being another popular method for representation [12]. Testing the interfaces with a wider range of graph representations and task types would ensure that results are more generalizable.

It is also important to remark that we also made specific design decisions in the implementation of the visualization interfaces, usually to support fair comparison between techniques. For example, only four small multiples were visible at a time and scrolling was required, and the most basic form of interactive animation was used whereby individual events are controlled with a slider. Similarly, we used linear interpolation in our animations; staged animated transitions could be considered [8, 24]. The effects of some of these secondary design decisions have the potential to be important, but are out of scope for this work. Future work manipulating further parameters will be welcome and should compare and further extend our findings.

It is possible that some of the advantages and disadvantages that we observed are significantly affected by the type of input (e.g., direct touch vs. indirect mouse). Although we have justified testing with touch input as the more natural way to work on large collaborative displays, indirect inputs such as computer mice or touch pads are probably still a more common way to interact with dynamic network visualizations. Hence, it is important to further investigate the role of input type in these results. We are already working on an experiment to empirically verify possible differences caused by input.

Similarly, the size of the display or the portion of the field of view that it covers could explain some of the differences that we observed. Investigating how performance with these interfaces varies will provide generalizability, but it could also offer valuable insights to design improved variants that are even better.

11 Conclusion

In this paper, we present the results of a series of experiments intended to formally evaluate methods to visualize dynamic networks on large touch displays. We were primarily interested in comparing interfaces for tasks that involve cross-time operations to see network structure and attribute variation. Two of our selected interfaces, interactive animation and small multiples, are already well studied in previous literature [4, 5, 21, 26, 51]. Our third selected interface, interactive timeslicing, has not previously been experimentally evaluated.

For tasks involving comparison of specific time points, interactive timeslicing offered greater speed than interactive animation or small multiples; when the comparison was of network structure (collections of edges), there was also an important difference in accuracy. In several instances, small multiples was also better than interactive animation.

For navigating time intervals, small multiples is faster than both interactive animation and interactive timeslicing when time intervals are larger (the far condition). Small multiples has the advantage that it naturally represents contiguous time intervals, whereas interactive timeslicing requires the participant to create and move each of these timeslices individually. Interactive timeslicing was also ranked by participants as the worst interface for completing tasks of this type. Whilst completing an exploration of time intervals some participants struggled with 'fat finger' [56] problems; time windows would be accidentally re-sized rather than moved. Further refinement of interactive timeslicing would help resolve the issues of distinguishing between these interactions.

Due to the lack of existing evaluations for interactive timeslicing, assessing interface performance with solo participants is a necessary starting point. However, large displays are commonly used in collaborative settings and previous evaluations have shown the value of collaboration for graph exploration [50]. With this in mind, future experiments evaluating the usability of these interfaces for collaborative situations is vital.

Acknowledgments

We would like to thank the EPSRC grant EP/N509553/1. This work was partially supported by a Microsoft Surface Hub award.

REFERENCES

[1] J. Ahn, C. Plaisant, and B. Shneiderman. A task taxonomy for network evolution analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):365–376, 2014. doi: 10.1109/TVCG.2013.238

[2] C. Andrews and C. North. The impact of physical navigation on spatial organization for sensemaking. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2207–2216, 2013. doi: 10.1109/TVCG.2013.205

[3] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. doi: 10.1007/3-540-31190-4

[4] D. Archambault and H. C. Purchase. Can animation support the visualisation of dynamic graphs? *Information Sciences*, 330:495–509, 2016.

[5] D. Archambault, H. C. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, 2011. doi: 10.1109/tvcg.2010.78

[6] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A descriptive framework for temporal data visualizations based on generalized space-time cubes. *Computer Graphics Forum*, 36(6):36–61, 2017. doi: 10.1111/cgf.12804

[7] B. Bach, N. Kerracher, K. W. Hall, S. Carpendale, J. Kennedy, and N. Henry Riche. Telling stories about dynamic networks with graph comics. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '16)*, pp. 3670–3682, 2016. doi: 10.1145/2858036.2858387

[8] B. Bach, E. Pietriga, and J. D. Fekete. GraphDiaries: animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(5):740–754, 2014. doi: 10.1109/TVCG.2013.254

[9] R. Ball and C. North. Analysis of user behavior on high-resolution tiled displays. In *Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction*, pp. 350–363, 2005. doi: 10.1007/11555261_30

[10] R. Ball, C. North, and D. A. Bowman. Move to improve: promoting physical navigation to increase user performance with large displays. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pp. 191–200, 2007. doi: 10.1145/1240624.1240656

[11] M. Bastian, S. Heymann, and M. Jacomy. Gephi: an open source software for exploring and manipulating networks. In *Third International AAAI Conference on Weblogs and Social Media*, 2009.

[12] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. A Taxonomy and Survey of Dynamic Graph Visualization. *Computer Graphics Forum*, 36(1):133–159, jan 2017. doi: 10.1111/cgf.12791

[13] L. Besançon and P. Dragicevic. The continued prevalence of dichotomous inferences at CHI. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, p. 1–11, 2019. doi: 10.1145/3290607.3310432

[14] A. Bezerianos, F. Chevalier, P. Dragicevic, N. Elmqvist, and J.-D. Fekete. GraphDice: A system for exploring multivariate social networks. In *Computer Graphics Forum*, vol. 29, pp. 863–872, 2010. doi: 10.1111/j.1467-8659.2009.01687.x

[15] X. Bi and R. Balakrishnan. Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1005–1014, 2009. doi: 10.1145/1518701.1518855

[16] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. doi: 10.1109/TVCG.2011.185

[17] I. Boyandin, E. Bertini, and D. Lalanne. A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples. *Computer Graphics Forum*, 31(3):1005–1014, 2012. doi: 10.1111/j.1467-8659.2012.03093.x

[18] U. Brandes and M. Mader. A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In *Graph Drawing*, pp. 99–110. Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-25878-7_11

[19] U. Brandes and B. Nick. Asymmetric relations in longitudinal social networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2283–2290, 2011. doi: 10.1109/tvcg.2011.169

[20] U. Brandes and D. Wagner. Analysis and visualization of social networks. In M. Jünger and P. Mutzel, eds., *Graph Drawing Software*, pp. 321–340. Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-642-18638-7_15

[21] M. Brehmer, B. Lee, P. Isenberg, and E. K. Choe. A comparative evaluation of animation and small multiples for trend visualization on mobile phones. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):364–374, Jan 2020. doi: 10.1109/TVCG.2019.2934397

[22] R. C. Brown, T. Fischer, A. D. Goldwich, F. Keller, R. Young, and P. L. Plener. #cutting: Non-suicidal self-injury (NSSI) on instagram. *Psychological Medicine*, pp. 1–10, 2017. doi: 10.1017/s0033291717001751

[23] M. Burch, C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf. Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2344–2353, 2011.

[24] F. Chevalier, P. Dragicevic, and S. Franconeri. The not-so-staggering effect of staggered animated transitions on visual tracking. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2241–2250, 2014. doi: 10.1109/TVCG.2014.2346424

[25] M. Czerwinski, G. Smith, T. Regan, B. Meyers, G. G. Robertson, and G. Starkweather. Toward characterizing the productivity benefits of very large displays. In *Interact*, vol. 3, pp. 9–16, 2003.

[26] M. Farrugia and A. Quigley. Effective temporal graph layout: A comparative study of animation versus static display methods. *Information Visualization*, 10(1):47–64, 2011. doi: 10.1057/ivs.2010.10

[27] Y. Frishman and A. Tal. Dynamic drawing of clustered graphs. In *IEEE Symposium on Information Visualization*, pp. 191–198, 2004. doi: 10.1109/infvis.2004.18

[28] Y. Frishman and A. Tal. Online dynamic graph drawing. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):727–740, 2008. doi: 10.1109/tvcg.2008.11

[29] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991. doi: 10.1002/spe.4380211102

[30] R. Gallotti and M. Barthelemy. The multilayer temporal network of public transport in Great Britain. *Scientific Data*, 2(1):1–8, 2015. doi: 10.1038/sdata.2014.56

[31] P. A. Gloor and Y. Zhao. Tecflow-a temporal communication flow visualizer for social networks analysis. In *ACM CSCW Workshop on Social Networks*, vol. 6, p. 5, 2004.

[32] S. Greenland. Invited commentary: The need for cognitive science in methodology. *American Journal of Epidemiology*, 186(6):639–645, 2017. doi: 10.1093/aje/kwx259

[33] S. Greenland, S. J. Senn, K. J. Rothman, J. B. Carlin, C. Poole, S. N. Goodman, and D. G. Altman. Statistical tests, p values, confidence intervals, and power: a guide to misinterpretations. *European Journal of Epidemiology*, 31:337–350, 2016. doi: 10.1007/s10654-016-0149-3

[34] F. Guimbretière, M. Stone, and T. Winograd. Fluid interaction with high-resolution wall-size displays. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 21–30, 2001. doi: 10.1145/502348.502353

[35] S. Hachul and M. Jünger. An experimental comparison of fast algorithms for drawing general large graphs. In P. Healy and N. S. Nikolov, eds., *Graph Drawing*, pp. 235–250, 2006. doi: 10.1007/11618058_22

[36] S. Hadlak, H.-J. Schulz, and H. Schumann. In situ exploration of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2334–2343, 2011. doi: 10.1109/tvcg.2011.213

[37] N. Henry Riche, A. Bezerianos, and J.-D. Fekete. Improving the readability of clustered social networks using node duplication. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1317–1324, 2008. doi: 10.1109/tvcg.2008.141

[38] P. Isenberg, P. Dragicevic, W. Willett, A. Bezerianos, and J.-D. Fekete. Hybrid-image visualization for large viewing environments. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2346–2355, 2013. doi: 10.1109/TVCG.2013.163

[39] M. R. Jakobsen and K. Hornbæk. Up close and personal: Collaborative work on a high-resolution multitouch wall display. *ACM Transactions on Computer-Human Interaction*, 21(2):11:1–11:34, Feb. 2014. doi: 10.1145/2576099

[40] M. R. Jakobsen and K. Hornbæk. Is moving improving? some effects of locomotion in wall-display interaction. In *Proceedings of the Annual ACM Conference on Human Factors in Computing Systems*, pp. 4169–4178. Association for Computing Machinery, 2015. doi: 10.1145/2702123.2702312

[41] N. Kerracher, J. Kennedy, and K. Chalmers. The design space of temporal graph visualisation. In *EuroVis*, vol. 14, pp. 7–11, 2014. doi: 10.2312/

eurovisshort.20141149

[42] N. Kerracher, J. Kennedy, and K. Chalmers. A task taxonomy for temporal graph visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 21(10):1160–1172, 2015. doi: 10.1109/tvcg.2015.2424889

[43] U. Kister, K. Klamka, C. Tominski, and R. Dachselt. GraSp: combining spatially-aware mobile devices and a display wall for graph visualization and interaction. *Computer Graphics Forum*, 36(3):503–514, 2017.

[44] A. Lee, D. Archambault, and M. Nacenta. Dynamic network plaid: A tool for the analysis of dynamic networks. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2019. doi: 10.1145/3290605.3300360

[45] N. Masuda and P. Holme. Predicting and controlling infectious disease epidemics using temporal networks. *F1000prime reports*, 5, 2013. doi: 10.12703/p5-6

[46] W. Mullër and H. Schumann. Visualization methods for time-dependent data-an overview. In *Proceedings of the Winter Simulation Conference*, vol. 1, pp. 737–745, 2003. doi: 10.1109/wsc.2003.1261490

[47] M. Nacenta, U. Hinrichs, and S. Carpendale. FatFonts: combining the symbolic and visual aspects of numbers. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 407–414, 2012. doi: 10.1145/2254556.2254636

[48] T. Ni, G. Schmidt, O. Staadt, M. Livingston, R. Ball, and R. May. A survey of large high-resolution display technologies, techniques, and applications. In *IEEE Virtual Reality Conference*, pp. 223–236, 2006. ISSN: 2375-5334. doi: 10.1109/VR.2006.20

[49] M. D. Plumlee and C. Ware. Zooming versus multiple window interfaces: Cognitive costs of visual comparisons. *ACM Trans. Computer-Human Interaction*, 13(2):179–209, 2006. doi: 10.1145/1165734.1165736

[50] A. Prouzeau, A. Bezerianos, and O. Chapuis. Evaluating multi-user selection for exploring graph topology on wall-displays. *IEEE Transactions on Visualization and Computer Graphics*, 23(8):1936–1951, 2017. doi: 10.1109/tvcg.2016.2592906

[51] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko. Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1325–1332, 2008. doi: 10.1109/TVCG.2008.125

[52] S. Rufiange and M. J. McGuffin. DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2556–2565, 2013. doi: 10.1109/tvcg.2013.149

[53] P. Saraiya, P. Lee, and C. North. Visualization of graphs with associated timeseries data. In *IEEE Symposium on Information Visualization (InfoVis '05)*, pp. 225–232, 2005. doi: 10.1109/infovis.2005.37

[54] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, sep 2006. doi: 10.1109/tvcg.2006.166

[55] B. Shneiderman and C. Plaisant. Interactive visual event analytics: Opportunities and challenges. *Computer*, 52(1):27–35, 2019. doi: 10.1109/mc.2018.2890217

[56] K. A. Siek, Y. Rogers, and K. H. Connelly. Fat finger worries: How older and younger users physically interact with PDAs. In M. F. Costabile and F. Paternò, eds., *Human-Computer Interaction*, pp. 267–280. Berlin, Heidelberg, 2005. doi: 10.1007/11555261_24

[57] P. Simonetto, D. Archambault, and S. Kobourov. Drawing dynamic graphs without timeslices. In *Proceedings of Graph Drawing*, pp. 394–409, 2018. doi: 10.1007/978-3-319-73915-1_31

[58] P. Simonetto, D. Archambault, and S. G. Kobourov. Event-based dynamic graph visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 26(7):2373–2386, 2020. doi: 10.1109/tvcg.2018.2886901

[59] N. A. Streitz, J. Geißler, and T. Holmer. Roomware for cooperative buildings: Integrated design of architectural spaces and information spaces. In N. A. Streitz, S. Konomi, and H.-J. Burkhardt, eds., *Cooperative Buildings: Integrating Information, Organization, and Architecture*, Lecture Notes in Computer Science, pp. 4–21. Springer, Berlin, Heidelberg, 1998. doi: 10.1007/3-540-69706-3_3

[60] D. S. Tan, D. Gergle, P. Scupelli, and R. Pausch. Physically large displays improve performance on spatial tasks. *ACM Transactions on Computer-Human Interaction*, 13(1):71–99, 2006. doi: 10.1145/1143518.1143521

[61] D. S. Tan, D. Gergle, P. G. Scupelli, and R. Pausch. Physically large displays improve path integration in 3D virtual navigation tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 439–446, 2004. doi: 10.1145/985692.985748

[62] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: can it facilitate? *International Journal of Human-Computer Studies*, 57(4):247–262, 2002. doi: 10.1006/ijhc.2002.1017

[63] J. Walker, R. Borgo, and M. W. Jones. TimeNotes: A study on effective chart visualization and interaction techniques for time-series data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):549–558, 2016. doi: 10.1109/tvcg.2015.2467751

[64] Y. Wang, D. Archambault, H. Haleem, T. Moeller, Y. Wu, and H. Qu. Nonuniform timeslicing of dynamic graphs based on visual complexity. In *IEEE Visualization Conference Short Papers*, pp. 241–245, 2019. doi: 10.1109/visual.2019.8933748

[65] R. L. Wasserstein and N. A. Lazar. The ASA statement on p-values: Context, process, and purpose. *The American Statistician*, 70(2):129–133, 2016. doi: 10.1080/00031305.2016.1154108

[66] J. S. Yi, N. Elmqvist, and S. Lee. TimeMatrix: Analyzing temporal social networks using interactive matrix-based visualizations. *International Journal of Human-Computer Interaction*, 26(11-12):1031–1051, 2010. doi: 10.1080/10447318.2010.516722