Gemini^2: Generating Keyframe-Oriented Animated Transitions Between Statistical Graphics

Younghoon Kim*
University of Washington

Jeffrey Heer†
University of Washington

Abstract

Complex animated transitions may be easier to understand when divided into separate, consecutive stages. However, effective staging requires careful attention to both animation semantics and timing parameters. We present Gemini^2, a system for creating staged animations from a sequence of chart keyframes. Given only a start state and an end state, Gemini^2 can automatically recommend intermediate keyframes for designers to consider. The Gemini^2 recommendation engine leverages Gemini, our prior work, and GraphScape to itemize the given complex change into semantic edit operations and to recombine operations into stages with a guided order for clearly conveying the semantics. To evaluate Gemini^2's recommendations, we conducted a human-subject study in which participants ranked recommended animations from both Gemini^2 and Gemini. We find that Gemini^2's animation recommendation ranking is well aligned with subjects' preferences, and Gemini^2 can recommend favorable animations that Gemini cannot support.

Index Terms: H.5.2 [User Interfaces]: User Interfaces—Graphical user interfaces (GUI); H.5.m [Information Interfaces and Presentation]: Miscellaneous

1 Introduction

Data analysis and communication often involve multiple statistical graphics and transitions among them [2]. To convey the changes that occur in transitions, visualization researchers study and employ animation techniques that assess the effectiveness of animated transitions [19, 27] and their various strategies, such as staging [11], staggering [4], time distortions [6], and bundling (or path optimizing) [7, 29]. Animation practitioners have created compelling media conveying data-driven insights, e.g., the famous Hans Rosling's presentation using Gapminder Trendalyzer [23], and an interactive article explaining machine learning algorithms [30].

However, authoring animated transitions between statistical graphics is not trivial; animation authors must implement low-level movements by writing code that considers many design variations, such as staging and timing. Recently, researchers have facilitated this authoring experience using a non-programming interfaces [26] or domain-specific languages [9]. However, authors must still manually generate animation keyframes or specify changes by selecting low-level graphic components.

In 2020, Kim & Heer introduced Gemini [15], a recommender system that suggests staged animations given start and end charts. However, Gemini's expressiveness is limited, as it cannot express independent intermediate states to the start and end charts. In addition, the Gemini's staged animation representations can be complex for users or systems to modify since they are combinations of the start and end chart specs with an animation spec in the Gemini grammar.

*e-mail: yhkim01@uw.edu
†e-mail: jheer@uw.edu

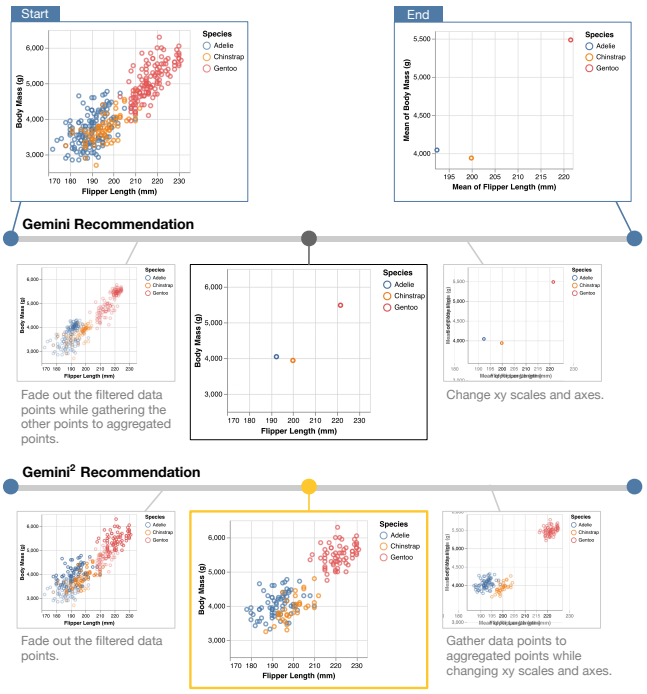

Figure 1: Staged animation timelines recommended by Gemini and Gemini^2 for a transition where data are filtered and aggregated. The gray lines represents timelines, and the circles indicate start, end, and intermediate states. Gemini^2 generates a keyframe chart (yellow circle) to separate two data transformations, which Gemini cannot do.

We introduce Gemini^2, a system for generating staged animated transitions between statistical graphics using keyframe sequences. Users first provide a chart array as a keyframe sequence. Then, Gemini^2 automatically generates animations connecting consecutive keyframes. Users can then create nuanced animations by adjusting the animation specifications that connect each adjacent keyframe pair. Gemini^2 thus supports a *keyframe-oriented animation authoring approach* [25], where users design animations by importing or demonstrating keyframes identified in visualization specifications and connecting them with animation specifications. We describe Gemini^2's enhanced expressiveness and ease-of-use vis-a-vis Gemini in Section 2.

Further, in Section 3 we demonstrate Gemini^2's utility by presenting a usage scenario that shows how it helps users create elaborate staged animations using keyframes. Additionally in this section, we evaluate keyframe recommendation through a crowd-sourced user study, where participants rank charts recommended by Gemini^2 and Gemini. We find that while Gemini^2's recommendations are similarly compelling as Gemini's, Gemini^2 can prioritize enumerated staged animations better than its predecessor. We conclude in Section 4 by discussing future directions, such as user interfaces, to improve the Gemini^2 animation authoring process.

2 Related Work

Animated transitions are commonly used to convey changes between two visualization states. Visualization researchers have shown that

animations can facilitate a range of visual analysis tasks, from low-level value comparison and object tracking to higher-level tasks such as assessing trends and distributions [3, 10, 12, 13, 18, 20, 21]. However, there is also justified skepticism of the effectiveness of animated transitions. Tversky et al. [27] found that some studies were biased in favor of animations, as the animation included additional information relative to other conditions. Further, they posed two high-level animation design principles, *Congruence* (the structure and content of the external representation should correspond to the desired structure and content of the internal representation) and *Apprehension* (the structure and content of the external representation should be readily and accurately perceived and comprehended). Robertson et al. [19] found that though animated time-series data in multiple small charts was less effective in analysis tasks, subjects preferred animation in a presentation context.

Prior research has proposed and studied various animation techniques to enhance communication capability, including timing variations [4, 6] and trajectory bundling [7, 29]. To convey complex changes, animations adopt staging techniques that divide the given changes into multiple sub-sequences and animate them consecutively. Staged animations have outperformed unstaged variants for value estimation and change tracking [11], conveying aggregate operations [14], and understanding online dynamic networks [5]. Gemini^2 aims to aid the authoring of staged animations by generating them automatically from given keyframes and by recommending effective keyframes for given start and end visualization states.

Thompson et al. [25] surveyed and characterize three animation authoring paradigms: *keyframe animation* (setting keyframes and tweening them consecutively), *procedural animation* (creating by behavioral parameters), and *presets & template animation* (applying pre-defined effects or templates). They found that most authors prefer keyframe animation, but animation authoring tools should support the other two paradigms depending on transition types and animation components. Gemini^2 facilitates the keyframe animation paradigm by letting users animate with customized keyframes and also by automatically suggesting keyframes.

A range of tools helps users create animated transitions, each making different trade-offs between *ease-of-use* and *expressiveness*. Focusing on ease-of-use, DataClip [1] provides pre-defined animation templates, and gganimate [22] lets users animate existing ggplot charts. Both limit expressiveness by using finite animation templates or making transitions within static visual encoding. In contrast, professional animation tools, such as D3 or Adobe After Effects, allow greater expressiveness but require significant effort, including writing program code to calculate and assign values for low-level components, diminishing ease of use. Between these extremes, researchers have developed declarative grammars for animated transitions, such as Canis [9] and Gemini [15]. These grammars help users avoid imperative programming while focusing on specifying animation designs. However, Canis still requires users to make low-level selections using W3C Selector syntax [28] (like D3).

In contrast, Data Animator [26] and CAST [8] enable keyframe animation authoring with a user-friendly graphical user interface (GUI). Users can set up keyframes by importing Data Illustrator [17] projects or selecting graphic components. Then, users can configure animation designs using timing parameters and data mappings between adjacent keyframes. Likewise, Gemini^2 lets users set keyframes by importing Vega-Lite visualizations. However, Gemini^2 focuses on helping initiate the authoring process via keyframe recommendations, but does not provide a GUI for a general authoring experience.

2.1 Comparison to Gemini

Gemini^2 enhances Gemini's expressiveness and ease-of-use. A limitation of Gemini is that its grammar cannot specify independent intermediate states to a start or end charts. For example, the intermediate states' data must be either the start or end chart's data. In addition, the intermediate states' representations cannot be isolated; it requires users or systems to combine both start and end visualization specs with an animation spec in the Gemini grammar. Gemini^2 instead permits intermediate state representations as independent visualization specifications, or keyframes. By doing so, Gemini^2 increases expressiveness for staged animations and enacts the keyframe animation authoring paradigm.

For recommendations, Gemini^2 incorporates GraphScape [16] to suggest intermediate keyframes for a given pair of start and end charts. GraphScape is a formal model allowing machines to analyze transitions between statistical graphics, such as itemizing transitions into semantic edit operations, and measuring cognitive costs to following transitions. We extend GraphScape to itemize and recombine a given transition into multiple intermediate keyframes (charts), and prioritize each path (from the start to end through the intermediate charts) using heuristic rules. We enable these extensions by allowing GraphScape to synthesize a chart by applying edit operations. With GraphScape's assistance, Gemini^2 can stage animations at a more fine-grained semantic level than Gemini can. For example, Gemini^2 can divide a transition with a filter and aggregate operations into two stages by separating them, but Gemini cannot since it treats such data transforms as a single data change of a mark component (Figure 1).

3 GEMINI^2

Gemini^2 extends the Gemini grammar to support a keyframe-oriented animation authoring process. It represents an animation design as a sequence of N keyframes (Vega-Lite [24] charts, denoted as k_i) and $N-1$ Gemini specifications(g_i) and is formally written as

$$Animation := \{(k_1, k_2, ..., k_N), (g_{1\to2}, g_{2\to3}, ..., g_{N\to N-1})\}$$

Keyframes are states that the animation passes through, and the Gemini specs are the specifications for each part of the animations connecting two adjacent keyframes. Therefore, users can express intermediate keyframes as arbitrary charts and connect them as an animation, which improves expressiveness relative to Gemini, as noted previously. Further, users can import multiple charts as keyframes and author staged animated transitions.

Gemini^2 generates animation designs using the existing Gemini compiler; it creates $N-1$ animation objects for each consecutive chart pair (k_i, k_{i+1}) and corresponding Gemini spec $(g_{i\to i+1})$. The compiled $N-1$ animation objects are wrapped as a single object that plays the animations in a row.

3.1 Gemini^2 Recommendations

Gemini^2 provides two recommendation features by extending GraphScape and Gemini: (1) *keyframe recommendations* for a given transition (start and end Vega-Lite charts), and (2) *animation recommendations* for a given keyframe sequence. These two recommendation features let users automatically generate staged animations for a given transition or keyframe sequence.

3.1.1 Keyframe Recommendations

For given starting and ending Vega-Lite charts, Gemini^2 can recommend intermediate keyframes to help authors explore staged animation designs. We implement these recommendations by leveraging GraphScape [16], a library for analyzing transitions between single-view Vega-Lite charts with Cartesian coordinates (e.g., bar, line, point charts). We extend GraphScape to generate charts; it now can synthesize charts by applying edit operations on existing charts. Also, we enable GraphScape to evaluate orders of edit operations in terms of how well they convey changes in an identifiable way.

GraphScape recommends keyframe sequences in three steps, as shown in Figure 2. It first itemizes changes from the start to the end into edit operations, which are atomic units that can be combined to

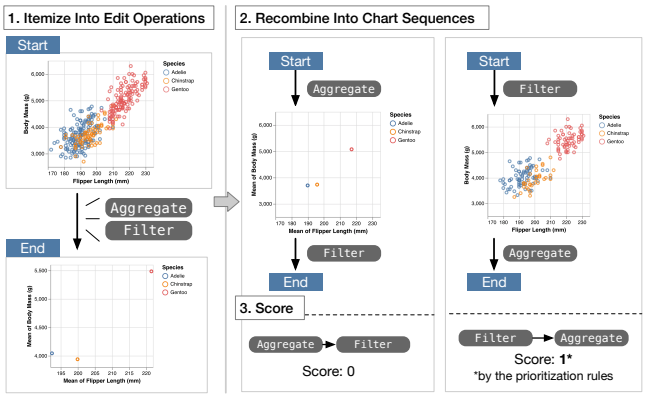

Figure 2: GraphScape's chart sequence recommendation process for a given transition. For given start and end charts, GraphScape enumerates chart sequences by itemizing their changes into edit operations and recombining the operations. Then, it scores the sequences to rank them according to predefined prioritizing rules (see Table 1).

represent any transition. Then, GraphScape enumerates intermediate charts by recombining and applying the itemized edit operations. During enumeration, GraphScape modifies the intermediate charts' scale domains (i.e., ranges of data values for corresponding visual properties) as the union of the given two charts to prevent unnecessary scale changes.

For example, imagine a transition filtering out some data and aggregating the remaining data into mean values. The start and end charts would have different scale domains ($domain_B \subset domain_A$). If an intermediate chart were applied only at the filter operation, the remaining data would have a different scale domain ($domain_C \notin \{domain_A, domain_B\}$), and the chart sequence from A to C to B would change scale twice. However, by modifying $domain_C$ to $domain_A$, the chart sequence changes scale only once.

After enumerating chart sequences, GraphScape prioritizes them based on how the edit operations are recombined. GraphScape applies a set of heuristic rules to evaluate the sequences. Each rule has (1) a *condition* determining if certain edit operations should follow other operations or be placed together, and (2) a *score* indicating if a sequence should be promoted or demoted if its operation order meets the condition. We define the rules in Table 1. Our goal is to make each edit operation identifiable by avoiding occlusion and implicit changes (such as filtering out some raw data when they are aggregated in charts). We hand-tuned the rule scores based on our experience and leave more rigorous tuning (e.g., data-driven adjustment in conjunction with user studies) as future work. We do not use GraphScape's transition costs in this prioritization process, another topic of potential future work. Gemini^2 imports the GraphScape chart sequence recommendations as keyframes for staged animation.

3.1.2 Animation Recommendations with Keyframes

Gemini^2 can recommend animations for a given keyframe sequence (k_i) and number of stages (M). The formal representation is:

$$Anim.Recom: \{(k_1, k_2, ..., k_N), M\} \rightarrow (H_1, H_2, ...)$$

where $H_i = (g^i_{1\rightarrow 2}, g^i_{2\rightarrow 3}, ..., g^i_{N-1\rightarrow N})$. The recommendation process leverages Gemini's recommendation feature. Gemini^2 first gets staged animation designs for each consecutive chart pair through the Gemini recommendation feature

$$G_{i\rightarrow i+1} = (g^1_{i\rightarrow i+1}, g^2_{i\rightarrow i+1}, ...).$$

In turn, it conducts a cross-join across each pair's recommendations. The cross-join enumerates arrays of Gemini specifications (H_i), each of which consists of $N-1$ specifications for every pair. Gemini^2 includes only those where the total number of stages equals the given number M.

$$Candidates = \{H_i | \forall i, \sum_j stage(g^i_{j\rightarrow j+1}) = M \;\&\; H_i \in G_{1\rightarrow 2} \times G_{2\rightarrow 3} \times ... \times G_{N-1\rightarrow N+1}\}$$

The enumerated Gemini^2 animation specifications are ranked by using Gemini's complexity measure:

$$f_{\text{total complexity}}(H_i) = \sum_{j=1} f_{\text{complexity}}(g^i_{j\rightarrow j+1}).$$

Gemini's complexity measure is a proxy for human effort to follow the animated changes. Gemini recommends staged animation designs with lower complexity measures. Similarly, Gemini^2 uses the summation of the complexity as an evaluation metric. Doing so is still valid because (1) Gemini's complexity is designed to be summed across each stage, and (2) each consecutive chart pair forms a stage. As a result, Gemini^2 prioritizes the Gemini specification array with the lowest total complexity score.

Table 1: Pre-defined rules for prioritizing edit operation order.

Condition & Description (Score)
Filter → Aggregate/Bin Data should be filtered before being aggregated or binned as it is difficult to convey what points are being filtered out from an aggregate mark. Likewise, data should be disaggregated/unbinned before being filtered. (Score: 1)
Aggregate → Bin Data should *not* be aggregated before being binned since data are aggregated by bin groups. Likewise, data should *not* be unbinned before being disaggregated. (Score: -1)
Marktype → Aggregate Marktype should *not* be changed before data are aggregated since some marktypes cannot support raw (un-aggregated) data, such as `bar` and `line` in Vega-Lite. Likewise, data should *not* be disaggregated before changing marktypes. (Score: -1)
Add/Remove/Modify Encoding → Aggregate New encodings should be added before data are aggregated so that the aggregated distributions can be shown in the extended encoding. Likewise, data should be disaggregated before removing encodings. (Score: 1)
Modify Encoding with Scale When modifying an encoding channel, its corresponding scale changes should occur together so users can perceive these changes as a single field change. For example, changing an existing variable A on the x-axis to B while modifying its scale to log scale should occur synchronously so users can see A → log(B) instead of A → B → log(B). (Score: 1)
Multiple Filters Multiple filter operations should *not* take place together since viewers cannot separately identify each filter operation. (Score: -1)
Bin and Aggregate Bin should be conducted with aggregate operations. Otherwise, the binned data points cause occlusion. (Score: 1)

3.2 Usage Scenario

We now demonstrate how Gemini^2 lets users create animations with keyframes through an example scenario. Here, a virtual author, K, creates Kim et al.'s two versions of staged animations, *staged elaborate* and *staged basic*, conveying a median aggregate operation for a 1D data distribution [14].

K starts to author *staged elaborate* by demonstrating its keyframe sequence in Vega-Lite charts (Figure 3). Once the keyframes are set up, K uses Gemini^2 to automatically generate a basic staged animation through its recommendations for the keyframe sequence. Then, K finishes by editing a small amount of the generated animation

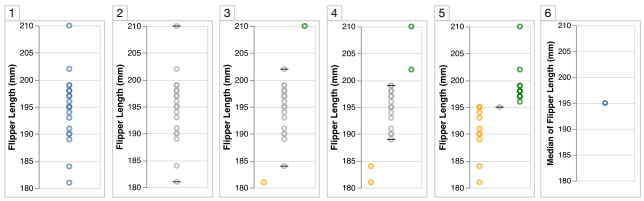

Figure 3: Example Gemini^2 keyframes for the staged elaborate animation for the median aggregate operation [14].

specifications to control the timing of each stage, including pauses and durations, plus a staggering effect on 4→5. The result can be simplified to a *staged basic* by removing keyframe 2, 3, and 4.

This authoring process is not available in Gemini since it cannot support defining intermediate states with extra mark layers (2 to 6) that were not included in start and end charts.

4 User Study

We conducted a user study to assess whether Gemini^2 recommended more compelling staged animation designs than Gemini.

We recruited 50 participants (35 males, 15 females) from Amazon Mechanical Turk. In the study, we provided 4 different types of transitions between statistical graphics. For each transition, subjects ranked 5 corresponding animation designs: two were Gemini's best recommendations for 2- and 3-stage animations (`g-s2`, `g-s3`), and two were Gemini^2's best recommendations for 2- and 3-stage animations (`g2-s2`, `g2-s3`) except for Stimulus 4. In Stimulus 4, Gemini^2's best is the same as the Gemini's best; therefore, we used 2- and 3-stage animations with lower keyframe recommendation scores to check if the score also aligned with subjects' preference. The fifth design was an unstaged animation (`no-stage`) that directly interpolates all changes. We included the unstaged animation to determine if Gemini^2 can recommend compelling staged animations for complex transitions better than the unstaged animations. More details are available on our supplementary material.

We analyzed collected users' ranking within each stimulus instead of merging them since Gemini^2 gave them different transition types and different animations rankings. We investigated animation designs' significant rank differences using the Friedman rank-sum test and Wilcox pairwise comparisons with Bonferroni corrections. Figure 4 shows bootstrapped means and 95% confidence intervals of subjects' ranking across Gemini^2's recommendation ranking.

Stimulus 1: Encoding & Data Transform & Marktype. The first stimulus set concerns a transition that modifies a given chart's x-axis variable, aggregates its y-axis value, and changes marktype from point to bar. The main difference between Gemini's and Gemini^2's staged animation designs was handling the legend: Gemini^2 staged the legend changes according to the intermediate keyframe, but Gemini did not. On average, subjects voted `g2-s3` as the best and `no-stage` as the worst. However, we found no significant differences among the animation rankings. Multiple subjects who ranked `no-stage` fifth reported that the animation "jumbled" the changes too quickly to understand how the data was transformed. This result suggests that Gemini^2 can recommend staged animations for complex transitions as well as Gemini can.

Stimulus 2: Two Filters. In this set, animations showed two filtering operations on a COVID-19 daily positive case chart; the first increased the time range of the chart, and the second introduced data from the other regions ("Other States"). Gemini^2's staged recommendations separated the data transforms into two steps: extending existing data for the new time range and introducing the other regions' data on top of the existing data. But Gemini's staged animations did not separate these data changes. Subject preferences are highest for `no-stage` but the difference is not significant difference. One possible interpretation is that these two specific filtering operations should be conveyed together since they resemble each other in terms of extending view; one extends regions, the other extends time range.

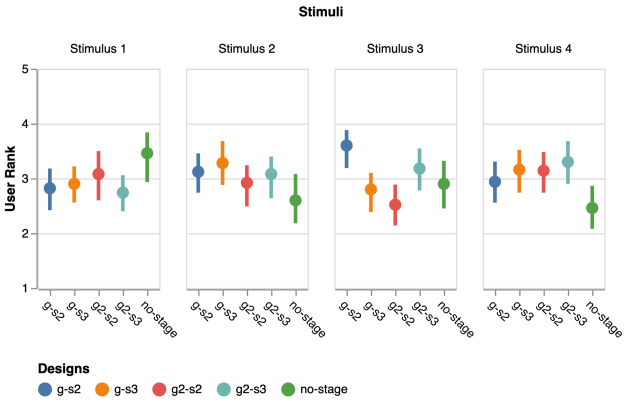

Figure 4: Bootstrapped means and 95% confidence intervals for each animation design per stimulus.

Stimulus 3: Encoding & Data Transform. This set involves transitions that add a y-axis for a new categorical field and aggregate data into means grouped by this field. Gemini recommended `g-s2` and `g2-s2` as the two best 2-stage animation designs. However, Gemini^2 picked `g2-s2` as the sole best 2-stage animation due to it prioritizing keyframe sequences that apply an add encoding edit operation before an aggregate edit operation. Participants ranked `g-s2` and `g2-s2` as the worst and the best, respectively. This result suggests that Gemini^2's keyframe sequence prioritization can distinguish staged animations that people may prefer.

Stimulus 4: Bin & Aggregate. The transitions in this set changed a 2D scatter plot to a 2D histogram by binning and aggregating. As previously mentioned, `g-s2` and `g-s3` were the best staged animation designs in Gemini^2 as well as in Gemini. Gemini^2 demoted `g2-s2` and `g2-s3` since they separated the bin and aggregated edit operations. The observed subjects' ranking seems to corroborate this prioritization, but none of the ranking differences are significant.

5 Conclusion and Future Work

In this work, we presented Gemini^2, a system for generating keyframe-oriented animated transitions between statistical graphics. We extended prior work on GraphScape and Gemini to make Gemini^2 represent animations using keyframe sequences and automate animations for a given transition or keyframe sequence. Gemini^2 is available as open-source software at `https://github.com/uwdata/gemini`.

Gemini^2 offers challenging and exciting future work opportunities. First, it is at present limited to represent animations among basic single-view Vega-Lite graphics supported by GraphScape. To support varied types of animated transitions, Gemini and GraphScape should be extended to handle more chart types. In terms of ease-of-use, one promising future direction is designing user interactions in animation authoring tools to exploit Gemini^2's recommendation features. Recently, Data Animator [26] introduced novel user interfaces to author keyframe animations between data graphics. The challenge is how to seamlessly blend Gemini^2's recommendation feature with such data graphic animation tools to help users explore alternative stagings while reducing their labor. Finally, additional animation perception studies might help improve Gemini^2's recommendation quality. Beyond observing users' preferences, measuring how well people understand or can identify changes through varied staged animation designs could help recommendation systems guide users toward increasingly effective designs.

Acknowledgments

NSF award IIS-1907399 supported this work.

REFERENCES

[1] F. Amini, N. H. Riche, B. Lee, A. Monroy-Hernandez, and P. Irani. Authoring data-driven videos with dataclips. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):501–510, Jan 2017. doi: 10.1109/TVCG.2016.2598647

[2] L. Battle and J. Heer. Characterizing exploratory visual analysis: A literature review and evaluation of analytic provenance in tableau. *Computer Graphics Forum*, 38(3):145–159, 2019. doi: 10.1111/cgf.13678

[3] B. B. Bederson and A. Boltman. Does animation help users build mental maps of spatial information? In *Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis'99)*, pp. 28–35, Oct 1999. doi: 10.1109/INFVIS.1999.801854

[4] 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, Dec 2014. doi: 10.1109/TVCG.2014.2346424

[5] T. Crnovrsanin, Shilpika, S. Chandrasegaran, and K.-L. Ma. Staged animation strategies for online dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):539–549, 2021. doi: 10.1109/TVCG.2020.3030385

[6] P. Dragicevic, A. Bezerianos, W. Javed, N. Elmqvist, and J.-D. Fekete. Temporal distortion for animated transitions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pp. 2009–2018. ACM, New York, NY, USA, 2011. doi: 10.1145/1978942.1979233

[7] F. Du, N. Cao, J. Zhao, and Y.-R. Lin. Trajectory bundling for animated transitions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 289–298. ACM, New York, NY, USA, 2015. doi: 10.1145/2702123.2702476

[8] T. Ge, B. Lee, and Y. Wang. *CAST: Authoring Data-Driven Chart Animations*. Association for Computing Machinery, New York, NY, USA, 2021.

[9] T. Ge, Y. Zhao, B. Lee, D. Ren, B. Chen, and Y. Wang. Canis: A high-level language for data-driven chart animations. *Computer Graphics Forum*, 39(3):607–617, 2020. doi: 10.1111/cgf.14005

[10] C. Gonzalez. Does animation in user interfaces improve decision making? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, pp. 27–34. ACM, New York, NY, USA, 1996. doi: 10.1145/238386.238396

[11] J. Heer and G. Robertson. Animated transitions in statistical data graphics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1240–1247, Nov. 2007. doi: 10.1109/TVCG.2007.70539

[12] J. Hullman, P. Resnick, and E. Adar. Hypothetical outcome plots outperform error bars and violin plots for inferences about reliability of variable ordering. *PLOS ONE*, 10(11):1–25, 11 2015. doi: 10.1371/journal.pone.0142444

[13] A. Kale, F. Nguyen, M. Kay, and J. Hullman. Hypothetical outcome plots help untrained observers judge trends in ambiguous data. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):892–902, Jan 2019. doi: 10.1109/TVCG.2018.2864909

[14] Y. Kim, M. Correll, and J. Heer. Designing animated transitions to convey aggregate operations. *Computer Graphics Forum*, 38(3):541–551, 2019. doi: 10.1111/cgf.13709

[15] Y. Kim and J. Heer. Gemini: A grammar and recommender system for animated transitions in statistical graphics. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):485–494, 2021. doi: 10.1109/TVCG.2020.3030360

[16] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer. Graphscape: A model for automated reasoning about visualization similarity and sequencing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pp. 2628–2638. ACM, New York, NY, USA, 2017. doi: 10.1145/3025453.3025866

[17] Z. Liu, J. Thompson, A. Wilson, M. Dontcheva, J. Delorey, S. Grigg, B. Kerr, and J. Stasko. Data illustrator: Augmenting vector design tools with lazy data binding for expressive visualization authoring. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, p. 1–13. Association for Computing Machinery, New York, NY, USA, 2018. doi: 10.1145/3173574.3173697

[18] B. Ondov, N. Jardine, N. Elmqvist, and S. Franconeri. Face to face: Evaluating visual comparison. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):861–871, Jan 2019. doi: 10.1109/TVCG.2018.2864884

[19] 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, Nov. 2008. doi: 10.1109/TVCG.2008.125

[20] G. G. Robertson, K. Cameron, M. Czerwinski, and D. C. Robbins. Animated visualization of multiple intersecting hierarchies. *Information Visualization*, 1:50–65, 2002.

[21] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pp. 189–194. ACM, New York, NY, USA, 1991. doi: 10.1145/108844.108883

[22] D. Robinson and T. L. Pedersen. gganimate. `https://gganimate.com/index.html`, 2018. [Online; accessed 29-July-2021].

[23] H. Rosling. The best stats youve ever seen. `https://www.ted.com/talks/hans_rosling_the_best_stats_you_ve_ever_seen`. [Online; accessed 29-July-2021].

[24] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, Jan 2017. doi: 10.1109/TVCG.2016.2599030

[25] J. Thompson, Z. Liu, W. Li, and J. Stasko. Understanding the design space and authoring paradigms for animated data graphics. *Computer Graphics Forum*, 39(3):207–218, 2020. doi: 10.1111/cgf.13974

[26] J. Thompson, Z. Liu, and J. Stasko. Data animator: Authoring expressive animated data graphics. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pp. 2628–2638. ACM, New York, NY, USA, 2017. doi: 10.1145/3025453.3025866

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

[28] W3C. W3c working draft. `https://www.w3.org/TR/css-transitions-1/`, 11 October 2018. [Online; accessed 29-July-2021].

[29] Y. Wang, D. Archambault, C. E. Scheidegger, and H. Qu. A vector field design approach to animated transitions. *IEEE Transactions on Visualization and Computer Graphics*, 24(9):2487–2500, September 2018. doi: 10.1109/TVCG.2017.2750689

[30] S. Yee and T. Chu. A visual introduction to machine learning. `http://www.r2d3.us/visual-intro-to-machine-learning-part-1/`, 2015. [Online; accessed 29-July-2021].