Visualizing Progressions for Education and Game Design

Eric Butler, Rahul Banerjee
Center for Game Science
Department of Computer Science & Engineering, University of Washington
{edbutler,banerjee}@cs.washington.edu

ABSTRACT
Progression design is a critical part of designing games or educational content. Currently, systems to visualize the content of a progression are limited and do not help designers answer questions important to the design process. These questions include comparing two progressions to understand the relative order in which concepts are introduced or how complexity changes throughout the progression. We present an interactive visualization system that allows designers to compare two different progressions, using multiple views and interaction techniques that aim to help designers answer these questions. We evaluate our tool through informal anecdotes, discussing insights that were found on progression data for actively developed games.

Author Keywords
data visualization; game design;

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION
A critical component of the design of educational or game experiences is crafting a coherent and effective sequence of content, called the *progression*. The progressions of skill-based games or educational curricula often aim to teach the player or student a series of different *concepts*. The design of progressions is a time-consuming, iterative process, where designers must carefully balance the ordering of concepts and how concepts are combined. More recently, designers and researchers have attempted to create systems that can produce such content automatically (e.g., [1]). Understanding the structure of these progressions and their properties is crucial for effective design of either hand-crafted progressions or progression-generating algorithms, and visualization tools can aid the designer in this process.

One important task is comparing the quality of two different progressions. Designs may be an in-development version of a game to a previous iteration, and deciding between two possible changes. In research settings, when developing systems to automatically generate progressions, it can be beneficial, for example, to compare the output of the algorithm to a human-designed progression. While their has been a lot of research into how to visualize player data [25], our task is slightly different: we are focused on looking at the static design space of the game, without any player data. There are several reasons designers may wish to analyze the design space of their game independent of player data. Often, early in the design process, developers do not have access to player data, or such data is too expensive to gather. There is value in getting "first impressions" of a design before putting it in front of players. Likewise, when experimenting with automatic progression generation techniques, researchers should be able to have some understanding of their systems without first having to collect player data.

However, there are few examples of such visualization systems, and such systems face several challenges. We are concerned specifically with understanding the progression of *concepts* that a player or student encounters in a skill-based game or piece of educational content. Thus, *progressions* are a sequence of *stages* (e.g., problems, puzzles, levels, study units), where each stage covers a different set of concepts. Both the number of concepts and number of stages may be high, in the dozens. So a progression is a sequence of high-dimensional data, presenting challenges in visualization. There are several activities we might want a useful progression comparison tool to support:

- What is the order concepts are introduced in two different progressions?
- How are particular concepts used in combination?
- How does the complexity change over the progression?
- How much of the "design space" is covered by once concept but not another; are there are there concepts or combinations that one of the progressions neglects?

Little work exists in visualizing this specific domain. Butler et al. visualize progressions as a table with stages on one axis and columns on the other [6]. This visual encoding does not support answering all the questions we care to support, especially when the number of concepts or stages are large. A good demonstration of how the table visualization fails to scale to a large number of stages/concepts can be found in Piotr Bugno's detailed outline of the story and puzzle progressions for *Portal 2*[1]. Though the ordering of concepts is clear, how they are combined is not, and viewing two of these

[1]`http://www.piotrbugno.com/2012/06/portal-2-timelines/`

charts together may yield little immediate insight as to their differences.

In this work we present a novel visualization system for comparing two different progressions to each other. We evaluate its usefulness through informal, anecdotal case studies using datasets of progressions from the game *Refraction*, and educational math game developed by our group. We discuss insights provided by this visualization tool that are difficult to notice without it.

RELATED WORK

The need to visualize sequences of values occur in a wide range of other domains, such stock market data or medical health records. There exist several techniques for visualizing scalar values that vary over time, like line and bar charts [22]. When comparing multiple scalar values over time, stacked area charts and their variants work well [12]. However, with higher-dimensional data (as ours is), effective visualizations must be created based on the semantics of the data. In domains ranging from personal histories [21] to treatment plans [15], such visualizations have been designed by hand, based on the specifics of the domain.

Visualizing high-dimensional data is challenging, because most display surfaces are two-dimensional. The use of 3D displays and/or animation can add at most two dimensions, but encoding the remaining dimensions in a visually perceptible way is challenging. Several encodings for such data have been explored. Some try to show all dimensions in the visualization. The simplest method is to plot all values in a table, which allows the sequential nature of the data to be clearly shown. However, some questions cannot be easily answered with tables, so we wish to supplement this view with other representations. Multiple surveys of visualization of multidimensional data have been done [26, 5], while others survey visualization methods for time-dependent data [18]. Chernoff faces [7] attempt to encode several dimensions using parts of the human face. It and related techniques, in addition to difficulty with accurate perception, do not scale to large number of dimensions. Parallel coordinates [13] plot the different dimensions on a set of parallel lines, using lines to connect particular datapoints across the dimensions. However, parallel coordinates do not allow for the sequence to be easily represented with position encodings. Another approach is to use dimensionality reduction techniques [11], sacrificing detail to increase interpretability. Linear methods for doing this include Principal Component Analysis (PCA) [10], Random Projections [4], etc., while non-linear methods include Kohonen's Self-Organizing Maps (SOM) [14], and Multidimensional scaling (MDS) [16]. We use MDS in our tool, though other dimensionality reduction techniques could be explored in future work.

A related but distinct visualization problem is that of showing player data. Current visualization systems targeted towards helping game developers with the design process mainly target visualizing player data. A recent survey describes several such systems [25]. Our system, in contrast, deals with data of a progression design space itself. Sequences of player actions actually have a very similar structure to progressions: a player travels through a high-dimensional state space, and several systems aim to visualize this traversal in a 2D representation. Researchers have found success using SOM (e.g., [9, 23]), MDS (e.g., [2]), or other layout techniques (e.g., [24]). However, one of our design requirements is to be able to drill down deeply into the set of concepts, for example, by looking at the ordering of concepts throughout the course of the progression. While we can use some of these techniques for our problem domain, existing systems for looking at player data do not cover these use cases.

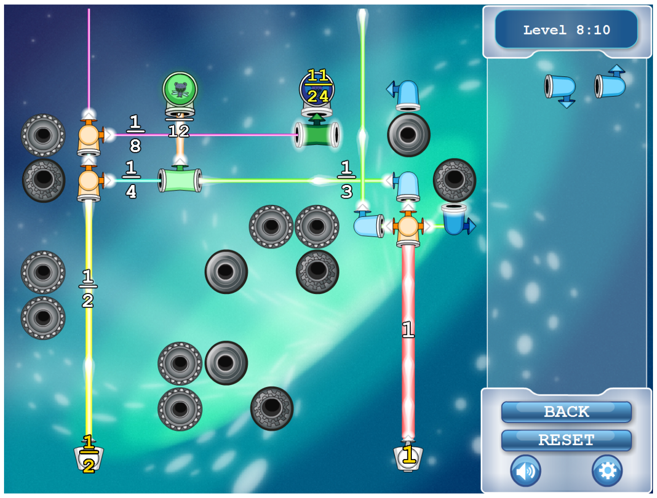


Figure 1: A stage of *Refraction*, the game from which most of our datasets are drawn. We continue to develop new progressions for the game, hence the need for design tools. Most of the *concepts* in *Refraction* describe the various mathematical or spatial challenges in each puzzle.

SYSTEM DESCRIPTION

Data Format

Here, we describe our representation of *progression* data, which are as a sequence of stages and concepts associated with them. Our system compares two progressions of a particular *game*. The data provides a set of *concepts*, $C = \{c_1, \ldots, c_m\}$, which represent various skills that appear in the game. The progression is a sequence of stages $s_1, \ldots, s_n$, and each *stage* s_i is described by a feature vector $s_i = \langle w_{i1}, w_{i2}, \ldots, w_{im} \rangle \in \mathbb{R}^m$, where w_{ij} describes "to what degree" or "how much of" concept c_j appears in stage s_i.

We assume that set set C is semantically meaningful, such that differences in "concept space" correspond to differences in how players will experience the game. In particular, we assume that the Euclidean norm of the difference between two stages captures some useful aspect of how different two stages are with regards to their contents. The designer must produce such a set to use the visualization system.

Refraction

We first give a brief description of the particular game from which most of are examples are drawn, and use it to illustrate concrete examples of progression *concepts*. *Refraction* is an educational puzzle game involving bending, splitting and combining lasers with fractional values. It combines mathematical and spatial reasoning challenges. A sample puzzle is shown in Figure 1. Much of our research work involves or requires creating different progressions (both human-made and computer-generated) for *Refraction* and other games, which is a primary reason tools to compare progressions would be useful (e.g., [6, 19, 20, 3, 17]).

Concepts in the game describe the various mathematical and spatial challenges of each puzzle. For example, the "`num_splitter`" concept describes how many *splitter pieces*, pieces that split lasers into equal fractional parts, must be used to solve the puzzle. The concept "`source_power_fraction`" describes whether *source pieces*, the pieces from which the lasers originate, emit lasers with fractional values or integer values. The datasets have two alternative concept sets; one is a full set of approximately 60 binary concepts, and the other is a set where those 60 concepts are collapsed into around 15 integer-valued categories. Other games have their own concept sets specific to their mechanics and rules.

Description of all the stuff in our tool

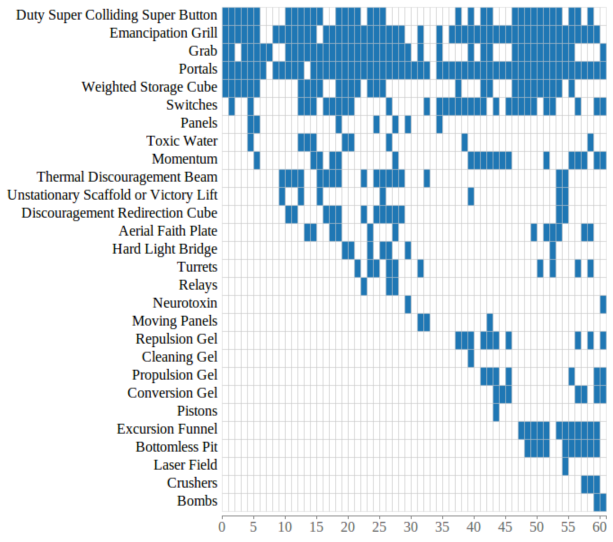
Our system provides three different views of progression data, and interactive controls for interacting with these views.

The *grid* is a table with concepts mapped to rows and stages mapped to columns. The entry at (i, j) represents the number of times that concept i appears in stage j (higher counts map to more saturated color). Figure 2 shows grid views for concept progressions from two different games (Refraction and Portal 2). When viewing the grid, our system lets the user sort the rows (concepts) according to their order of appearance . Thus, the concept introduced first in the progression is on the top row, and the one introduced last is assigned the last row.

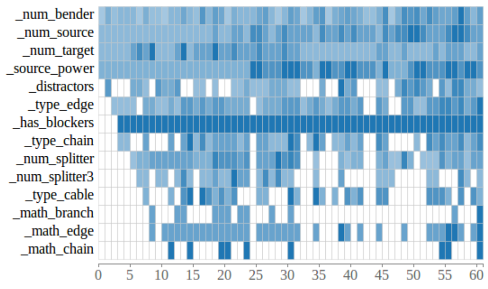
The *projection* view shows MDS projections for progressions, computed using the SMACOF (Scaling by Majorizing a Complicated Function) method [8]. Concept occurrence counts from each stage (column) form a k-dimensional vector $\langle c_1, c_2, \ldots, c_k \rangle$, and these vectors are projected into 2 dimensions, which are mapped to position. Every stage is rendered as a node, and nodes corresponding to adjacent stages are connected via edges. Multiple progressions are drawn in the same 2D region, rendered with transparency to mitigate occlusion caused by overlaps.

The user can also select a particular set of concepts and "drill down" to see how their occurrences vary precisely over various stages by selecting them in the "Filter To:" control. When one or more concepts are selected in the filter, our system shows *bar charts*, one chart per concept. The concept occurrence counts are mapped to the bar heights, with different stages on the horizontal axis. When no concepts are selected (or the "Clear Filters" button is clicked), the bar charts are hidden and the grid view is shown.

Brushing is supported inside all views by clicking and dragging to specify a rectangular region, which selects all elements inside its extents. Selection changes an element's color to red. Linking ensures that brushing inside the grid or bar chart view automatically selects the corresponding elements in the projection view (and vice-versa).



(a) Portal 2



(b) Refraction

Figure 2: Concept progressions for (a) Portal 2 and (b) Refraction displayed as grids.

Figure 3 shows a screenshot of our system being used to compare two progressions for Refraction. On the left are sorting controls (used for ordering the rows in the bar chart), and filters below (to select particular concepts for inspection). Here, two concepts (_num_bender and _num_splitter) are selected. In the center are bar charts showing the concept occurence counts for the selected concepts (one set of bar charts per progression). On the right, we see the MDS projection for both progressions, with some nodes selected via brushing, and corresponding nodes in the bar charts selected via linking.

DISCUSSION

In this section we discuss anecdotes of insights made possible through use of this too as well as limitations of the system.

Recent research efforts in our group have attempted to create progressions (for games such as *Refraction*) automatically. One goal of these algorithms was to replicate the success of the human-designed progressions, and one approach is to try to create progressions with the same basic structure as

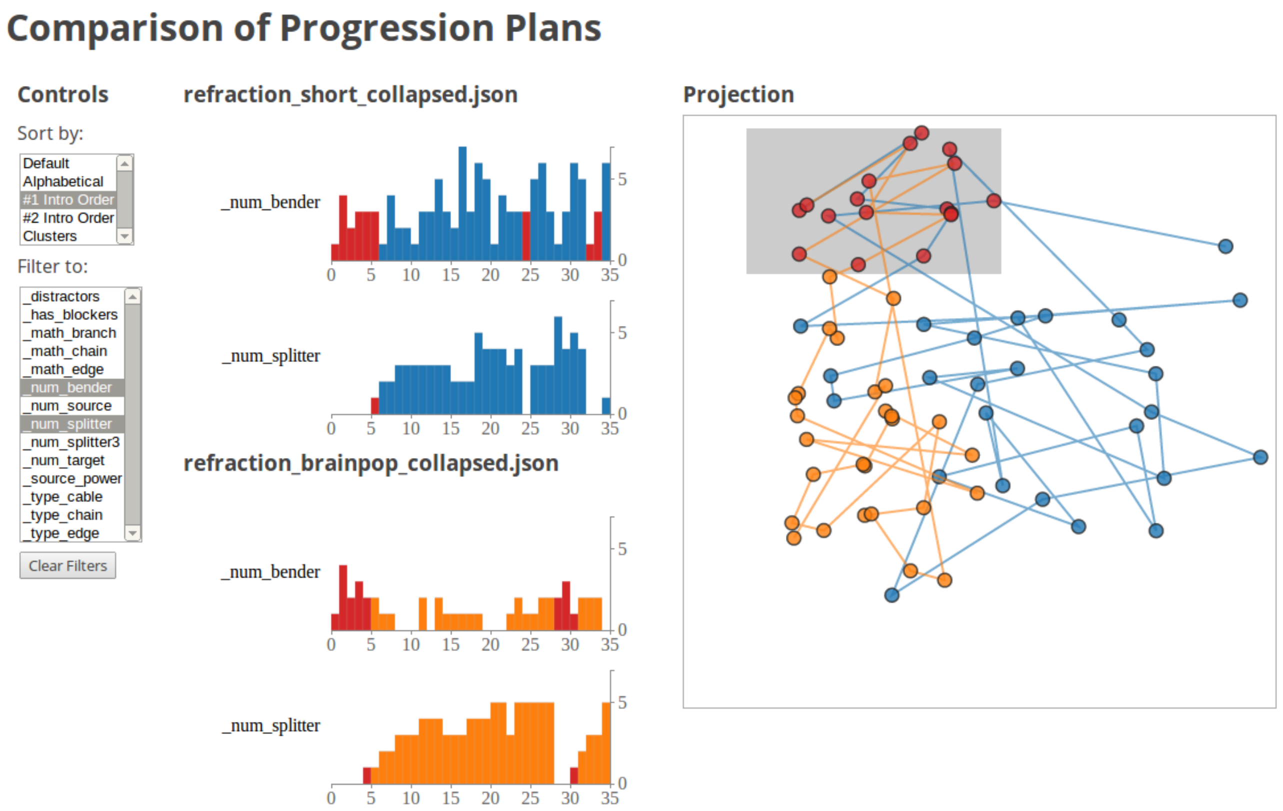


Figure 3: A screenshot of our system's interface, showing its filtering, brushing and linking capabilities. Here, on the left is a detailed view of two particular concepts for each progression, while the right shows the MDS projection of both progressions onto a 2D plane.

human-designed ones. When creating these algorithms, the best tools we had were manual inspection by playing through the game or looking at tables of computed features. However, when a computer-generated and a human-designed progression were compared in our tool, certain strong differences were immediately apparent. Figure 4 shows the MDS projection of two progressions: blue is human, orange is computer. It is clear that the successive stages in the computer progression are very close in "concept space," whereas the human progression moves much more chaotically through the space. This leads us to infer that the computer-created progression does not have enough variance from one puzzle to the next. Another interesting feature that is immediately obvious is that the computer progression eventually gets stuck and gives essentially very similar puzzles 20 times in a row.

Another measure on which we want to compare the progressions is ordering of concepts. The tools allows us to easily sort one progression by the ordering of another, in order to make differences readily apparent. This is illustrated in Figure 5, where we compare the same progressions by ordering. Here, we use a very detailed breakdown on the concept space. We immediately notice several concepts that appear very early in the computer-generated progression but very late in the human-generated progression. Additionally, we can clearly see how the games diverge in the later half of the progression. The sets of concepts introduced in the second half of each progression are nearly disjoint.

Limitations

One significant limitation in the system is that is relies on the concept set to be semantically meaningful so that operations such as taking the Euclidean distance between stages is sensible. This is obviously a tall order for game designers to satisfy before they can use the tool, as it is not clear which concepts actually affect player experience without testing. One possible remedy for this is to explore unsupervised machine learning techniques to deduce more meaningful concept dimensions, perhaps by clustering of dimensionality reduction techniques. If this tool were being used in a later stage of development when player data from user tests are available, that could be used in such an analysis by trying to learn which particular concepts have measurable impact on player experience. Another approach would be to allow the designer to interactively organize and categorize concepts while exploring with the tool, rather than beforehand.

Another major problem is the lack of a significant number of datasets for this domain. Though we believe this will change in the future, currently, explicitly creating data representations of progressions is rare and limited primarily to a small set of research groups. While we can (and did) manually

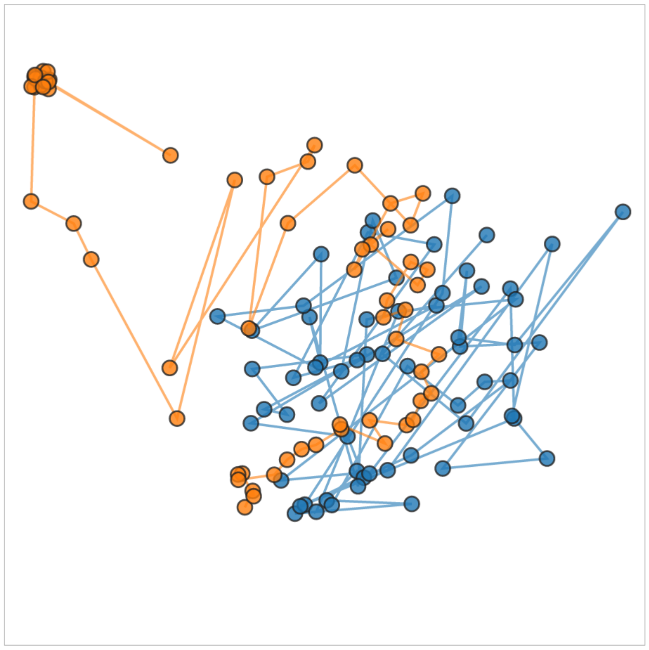


Figure 4: A human-crafted progression (blue) and one of our attempts at an automatically-generated progression (orange). The MDS projection clearly illustrates some of the qualitative differences between the two progressions.

create datasets for existing games, games very rarely have multiple progressions in final versions, as multiple progressions often only appear during iteration in the development process. On the other hand, while educational domains very frequently have wildly different progressions for a particular domain (e.g., high-school algebra), formal representations of these progressions for use in visualization are very rare. Therefore, though we believe these activities can be generally applicable to a wide range of games or educational domains, it is difficult to substantiate such a claim without other datasets on which to evaluate the tool's generality.

A minor limitation of our system is that is currently does not scale gracefully to extremely long progressions. While progressions with around 100 stages display quickly, MDS takes a long time on significantly longer systems, increasing the time users must wait before being able to interact.

CONCLUSION

We have presented a system that aims to help designers compare and answer important design questions about progressions during the design process. The tool supports several different encodings of progression data such as tables and MDS projections, supporting answering questions about things such as the ordering of introduction of concepts or how two different progressions move through the design space. Anecdotal evidence suggests that this tool can be useful in providing insight to designers.

There are several avenues of future work, in addition to overcoming previously mentioned limitations. We only used one dimensionality-reduction technique; others should be explored to find more effective ones. Though our system is designed for use where player data is not available, it could be improved in cases where such data does exist by integrating visualizations of such data into the tool. For example, the tool could allow designers to drill down into a particular stage and view statistics on how players performed on that stage, or aggregate statistics about each stage could be displayed alongside the table. Though this system was designed for a research setting, industry developers could likely benefit from such a tool, and a more thorough evaluation of the requirements for such a tool could be conducted. This system supports linear sequences; however, several professional or research games or educational systems use non-linear progression structures, for example by choosing the next math problem based on performance of the student. Techniques to better understand these structures could be explored.

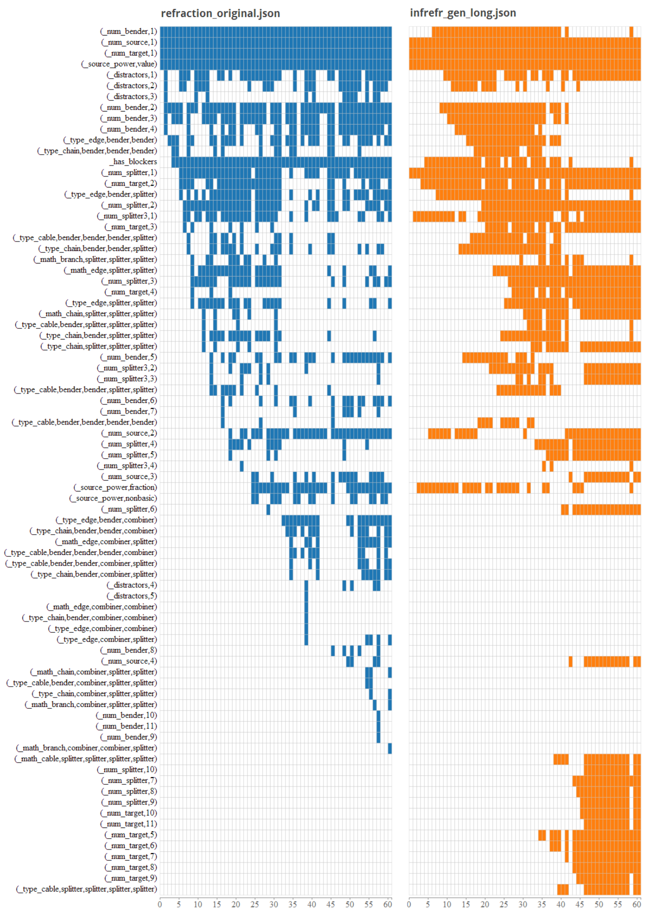


Figure 5: Comparing ordering of concepts between human-crafted (blue, on left) and computer-generated (orange, on right) progressions. The concepts are sorted by introduction order of the left progression, making differences in ordering between the two datasets apparent. Note that while typically this grid view uses hue to show quantitative values for each concept, this particular concept set is all binary. This screenshot is cropped for display purposes; the tool shows it slightly differently.

REFERENCES

1. Andersen, E., Gulwani, S., and Popović, Z. A trace-based framework for analyzing and synthesizing educational progressions. In *CHI* (2013).
2. Andersen, E., Liu, Y.-E., Apter, E., Boucher-Genesse, F., and Popović, Z. Gameplay analysis through state projection. In *FDG* (2010).
3. Andersen, E., O'Rourke, E., Liu, Y.-E., Snider, R., Lowdermilk, J., Truong, D., Cooper, S., and Popović, Z. The impact of tutorials on games of varying complexity. In *CHI* (2012).
4. Baraniuk, R. G., and Wakin, M. B. Random projections of smooth manifolds. *Foundations of computational mathematics 9*, 1 (2009), 51–77.
5. Buja, A., Cook, D., and Swayne, D. F. Interactive high-dimensional data visualization. *Journal of Computational and Graphical Statistics 5*, 1 (1996), 78–99.
6. Butler, E., Smith, A. M., Liu, Y.-E., and Popović, Z. A Mixed-Initiative Tool for Designing Level Progressions in Games. In *UIST* (2013).
7. Chernoff, H. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association 68*, 342 (1973), 361–368.
8. De Leeuw, J., and Mair, P. Multidimensional scaling using majorization: SMACOF in R.
9. Drachen, A., Canossa, A., and Yannakakis, G. N. Player modeling using self-organization in tomb raider: Underworld. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, IEEE (2009), 1–8.
10. Dunteman, G. H. *Principal components analysis*. No. 69. Sage, 1989.
11. Fodor, I. K. A survey of dimension reduction techniques, 2002.
12. Havre, S., Hetzler, E., Whitney, P., and Nowell, L. Themeriver: Visualizing thematic changes in large document collections. *Visualization and Computer Graphics, IEEE Transactions on 8*, 1 (2002), 9–20.
13. Inselberg, A., and Dimsdale, B. Parallel coordinates. In *Human-Machine Interactive Systems*. Springer, 1991.
14. Kohonen, T. *Self-organizing maps*. Springer, 2001.
15. Kosara, R., and Miksch, S. Metaphors of movement: a visualization and user interface for time-oriented, skeletal plans. *Artificial intelligence in medicine 22*, 2 (2001), 111–131.
16. Kruskal, J. B., and Wish, M. *Multidimensional scaling*. Sage, 1978.
17. Mandel, T., Liu, Y.-E., Levine, S., Brunskill, E., and Popovic, Z. Offline policy evaluation across representations with applications to educational games.
18. Mullër, W., and Schumann, H. Visualization methods for time-dependent data-an overview. In *Simulation Conference, 2003. Proceedings of the 2003 Winter*, vol. 1, IEEE (2003), 737–745.
19. O'Rourke, E., Butler, E., Liu, Y.-E., Ballweber, C., and Popovic, Z. The effects of age on player behavior in educational games. *Foundations of Digital Games, FDG 13* (2013).
20. O'Rourke, E., Haimovitz, K., Ballwebber, C., Dweck, C. S., and Popovic, Z. Brain points: A growth mindset incentive structure boosts persistence in an educational game.
21. Plaisant, C., Milash, B., Rose, A., Widoff, S., and Shneiderman, B. Lifelines: visualizing personal histories. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (1996), 221–227.
22. Playfair, W. *Playfair's commercial and political atlas and statistical breviary*. Cambridge University Press, 2005.
23. Thawonmas, R., Kurashige, M., Iizuka, K., and Kantardzic, M. Clustering of online game users based on their trails using self-organizing map. In *Entertainment Computing-ICEC 2006*. Springer, 2006, 366–369.
24. Wallner, G., and Kriglstein, S. A spatiotemporal visualization approach for the analysis of gameplay data. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, ACM (2012), 1115–1124.
25. Wallner, G., and Kriglstein, S. Visualization-based analysis of gameplay data–a review of literature. *Entertainment Computing 4*, 3 (2013), 143–155.
26. Wong, P. C., and Bergeron, R. D. 30 years of multidimensional multivariate visualization. In *Scientific Visualization* (1994), 3–33.