Exploring Player Behavior with Visual Analytics

Michael Eagle, Matthew Johnson, Tiffany Barnes, and Acey Boyce
University of North Carolina at Charlotte
9201 University City Blvd
Charlotte, NC, USA

ABSTRACT

In designing puzzle and educational games, it is critical to be able to understand player behavior, in order to provide feedback when a player needs help, or redesign a game to keep players on-task. However, building a system that can react to all possible player behaviors can be very time intensive, and if a redesign is needed, can be a wasted effort. We propose a novel visual analytic approach to analyzing playtest data to help the game design process, and demonstrate its application to BeadLoom Game. The approach helped the game developers identify uninterested players, and refine the game so players could get a better sense of how close they were to the puzzle's solution.

Keywords

Data Visualization, Behavior Networks, ITS, EDM

Categories and Subject Descriptors

K.8 [**General**]: Games

General Terms

Design

1. INTRODUCTION

Designing games is a complex task, and players are increasingly expecting highly individualized game experiences. One common method for tuning the player experience is play-testing, where players try the game before its release, exposing glitches in gameplay and difficulty in both surveys and game-play logs. However, effectively using the large amount of data available from a playtest can be challenging. Larger game companies such as Microsoft Studios have developed their own data analytics tools to understand these large complex player interaction datasets. In games set in three dimensional environments, these tools act like geographic information tools, overlaying player trace data on a game's level maps. However, in puzzle games and many educational games, there is a need to visualize the sequence of player behavior, but there is no logical spatial way to overlay player behavior over the usual visual representation of the game - except perhaps as a video. This makes it much more challenging to understand how, when, and where a puzzle or educational game design may need improvement.

We have developed InVis, short for Interaction Visualization, a visual analytics tool to help designers visualize sequential game states for a large number of playtesters at once. InVis was created to provide software developers and designers with new insights into the behavior of their players and their in-game experiences.

InVis was designed to visualize log-data from intelligent tutoring systems - software built to support learners in interactive problem solving. At the core of InVis is the idea of visualizing the "state space" through a behavior network. All game developers have an idea of a game's state - an inventory of the current values for all variable aspects of a game. In educational software and in puzzle games, this state can correspond to where someone is in a problem-solving sequence - and can often be understood by taking a simple screen shot. In both intelligent tutoring systems and in puzzle games, the experience can be individualized with feedback to direct attention to mistakes, or provide hints on what might be done next.

We believe InVis can support the iterative development and testing of puzzle and educational games, by helping designers visually analyze player behavior to identify ways to better craft the player experience. We demonstrate how we have used InVis to explore game-log playtest data from the BeadLoom Game, a puzzle game designed to teach math concepts. Our case study shows that InVis helped developers discover surprising player behaviors, identify bugs and inefficient aspects of the interface, and design potential algorithms for individualizing the game experience.

InVis can be used to understand user behavior in 2D and sequential problem solving environments; allowing investigators to view large numbers of player behaviors at once. This can be helpful in understanding player interactions with data, the HUD, or 2D puzzle solving, providing important insights that can be used for iterative improvement of a game. This work is useful for game designers interested in observing the ways players are interacting with their games, educational researchers who want to evaluate learning in game environments, and developers who want another way to debug player experiences.

2. RELATED WORK

Intelligent Tutoring Systems and video games are similar in that both incorporate rapid-feedback loops and scaffolding techniques [2]. Recent research has looked for ways to leverage the ITS and video game communities for future learning environments [8]. Typically an intelligent tutoring system is designed to provide a unique educational experience for each student. To provide this personalized experience, intelligent tutors use a student model and from the interactions logged by the user, map a user to some student model, then make assumptions about that user based on the model as is done by both the Hint Factory [9] and Bayesian Knowledge Tracing[7]. These same techniques can be used in games to tune player experiences as well. Mining educational data produced from Intelligent tutoring systems has helped develop models on specific student behaviors such as off-task behavior and gaming [2]. Visualizing interaction data from tutoring systems or games can be used to improve student models.

Noobler uses visualization to enhance the quality of the "spectator mode" for multi-player first-person shooters, incorporating many different visualization techniques to show a more complete overview of the action taking place in the game [5]. Similar to Moura et. al [6] our interest is in visualizing and understanding player behavior, with the focus on states and transitions, rather than position and spatial information.

In games research, Andersen analyzed game data to create game "states" - snapshots of what has been done in a game, and applied different metrics to game-data to assign a value to each of these states [1]. Using these values, Andersen created a dissimilarity matrix, which stores the difference, or distance, between each state-pair, and then used classical multidimensional scaling to project those relative distances to 2D space. This approach works well for visualizing state similarity, but does little to show the overall behavior of the players. It also does not present the information in a way that visually preserves sequence information.

2.1 Visualizing Interaction Data

InVis is a visualization tool for providing educators, researchers, and software designers the ability to explore data-logs in order to better understand how students or players are using software. Ideally InVis facilitates player behavior discovery and for interested persons hypotheses generation and confirmation about how their players interact.

In InVis evaluating and representing sequences of actions and states that closely models the behaviors of players is important. States retain all the details of their dimensions easily visible through the use of details on demand features. Filtering options allow a user to focus their investigation. This allows designers to explore all the information and aggregated statistics gathered by their game-logs, providing an effective method for gaining insights into how players behave.

2.2 BeadLoom Game

The Culturally Situated Design Tools (CSDT) [4] were designed to teach mathematics from a cultural context. The BeadLoom Game (BLG) [3] is a game-based extension of the CSDT: Virtual BeadLoom. The BLG and the original CSDT are built to give their users experience and practice with Cartesian coordinates, with an intended audience of middle school aged children. The BLG added game elements to the Virtual BeadLoom in order to increase motivation and learning [3]. Students can use point, line and shape functions, along with color parameters to make images in a Cartesian grid.

3. BEHAVIOR NETWORK

The novel Behavior Network Model, can be used across many domains to describe sequences of user interactions in software interfaces. We have applied the Behavior Network model to two widely different domains (logic proofs and drawing pictures on a grid) to understand student behavior. We model a solution attempt as a sequence of states (vertices) and actions (edges). We use *case* to refer to a single student's solution attempt. We create the behavior network for a problem by conjoining the set of all of its solution attempts. We use *state* to describe the state of the software environment, representing enough information so the program's state could be regenerated in the interface. We use *actions* to describe user interactions and their relevant parameters. We also store the set of all cases who visited any particular state-vertex or action-edge, allowing us to count frequencies and connect case specific information to the behavior network representation. This representation results in a connected, directed, labeled multi-graph with states as vertices, directed action edges to connect the states, and cases that provide additional information about states and edges. This representation allows us to build a behavior network model from any system that logs interactions in state, action, resulting-state tuples.

In order to gain a better understanding of the BLG data, we used the InVis Tool to explore player solutions. The state representation is a 41x41 array containing the 12 color values from the BLG game. Actions are represented by the six available bead-placement tools, as well as the relevant parameters for each tool. We also store the set of all cases who visited any particular state-vertex or action-edge.

For the BLG data, action-order is not preserved in the state description. That is, if the player had reached the same state (same red square) by repeatedly using the point tool, the final states are equivalent.

4. CASE STUDY

We collected game log-files from a study performed on the BLG in 2010. Data came from a total of 6 classes, ranging from 6th to 8th grade; for a total of four sessions. There were 132 students, and 2,438 game-log files. The students were split into two groups (called A-day and B-day) and were presented with BLG features in different orders. The A-Day students were given access to custom puzzles (a free play option,) while B-Day students were given a competitive game element in the form of a leader board. In order to investigate whether or not there were different problem solving patterns between the groups, we colored vertices based on the percentage of students who visited from each group. The values were normalized from green (A-Day) to red (B-Day.) We then loaded the data into the InVis tool and presented it to the developers of the BeadLoom Game.

Next we met with the designer and a developer from the BeadLoom Game and asked them to explore their log data using InVis. We hypothesized that InVis would allow users to discover unexpected and surprising details about their

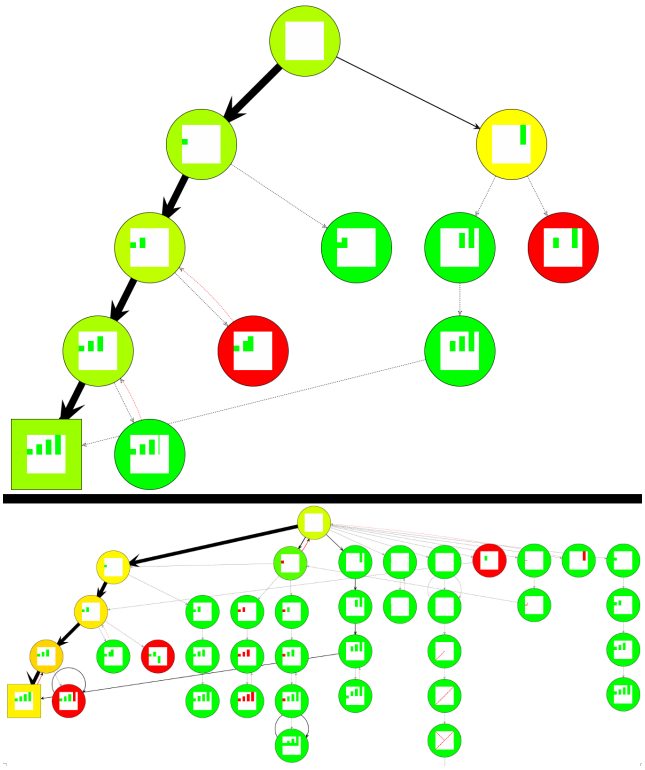

Figure 1: The top image is student attempts on the 3rd day; first-day attempts are shown on the bottom. As students become more familiar with the tool they solve the problem better, making fewer mistakes, and thus fewer states are visited. The goal state is represented as a square.

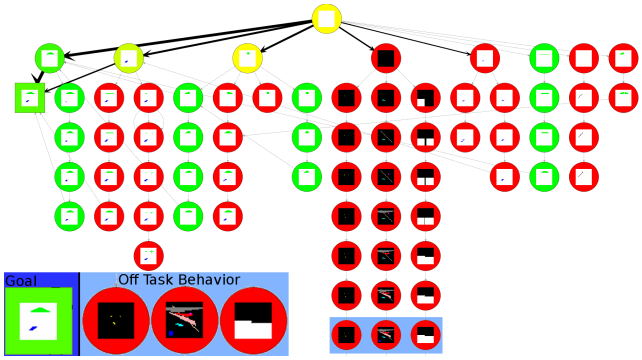

Figure 2: Off-task behavior is highlighted in this solution network. The final state of three students is highlighted and enlarged. It is clear the students were not working towards the actual goal, shown as a square node on the left side.

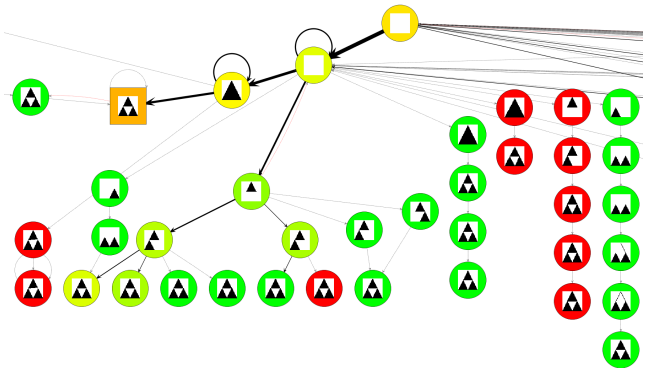

Figure 3: The goal image is in the orange box, while all other images are not recognized as the goal. Several states are close to the solution, but off by several pixels.

data; provide insights about player behavior; provide an efficient means to understand the general player behavior; and allow observers to identify 'strange' or outlying behavior.

InVis supports a variety of straight forward interactions which can be used to explore the data contained in a behavior network. Users can filter the network based on the data they load into the vertices, edges, or aggregate data like frequency. The mouse provides panning and zooming and selection, while GUI buttons provide the generation of sub-networks, which can be used to further zoom and filter a behavior network. Search options exist for finding specific players and the states and actions they visited and used respectively.

In figure 1 we have a set of students who worked on the same problem on two different days, the first and third day of the study. By looking at the number of states we can see a more diverse set of attempts on the first day. It is possible that the change in the number of states over time is the visual representation of learning, which this image might suggest, additional research will be necessary to determine if that is so. Additionally we considered the idea that perhaps vertex count could be used as part of a measure for determining problem difficulty.

In game off-task behavior was a common problem as seen in figure 2, with a number of students forgoing the goal and designing their own images in the game instead. Some form of this type of behavior was present in roughly 90 percent of the puzzles solved by the students. A free-play mode is supported with the BeadLoom Game and can be found in the custom puzzle section of the game, it is arguable that students found the step of changing modes an unnecessary one.

Figure 3 shows a potential flaw in the BLG, the states along the bottom of this image are almost indistinguishable from the goal vertex. Some of those solutions are only off by as much as 2 pixels. It seems to be due to the way the triangle algorithm works in comparison to the triangle iteration algorithm within the BLG. Identifying this case in the data has lead the developers to make two design changes to the game. The first is to allow the player to see a count of how many pixels or beads remain incorrect, as well as a pop up window to inform players when they have successfully completed a puzzle, rather than forcing the student to click a button on the user interface. Discovering this particular case in the data was a surprising and helpful discovery for the game's developers.

In figure 4 we can see a number of child states generated from a single state. The interesting feature of this image, is most of those states have a red returning action which represents undo. This state for whatever reason contained

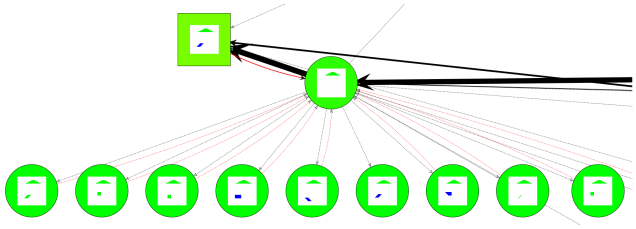

Figure 4: Each of the states on the bottom represents a student performing an incorrect move, and then undoing. This could perhaps be modeled by keeping track of states that are returned to frequently.

a number of mistakes made by students, but were quickly identified by the students.

5. DISCUSSION AND CONCLUSION

The designers were able to identify a variety of design changes they would like to make to the game after spending roughly 20 minutes using InVis. The most surprising detail the developers were interested in was the number of students who seemed to participate in off-task behavior, as seen in figure 2. Next was a design issue regarding automatic feedback, after viewing and exploring the Behavior Network of the BLG data, the developers recognized a number of people performed actions, even after arriving at the goal state. In the original game users were required to click when they were finished, however after discovering the undesirable behavior, an automated method was designed so the level will end once the goal-state is reached. There were also a number of cases where students created the goal-image, but it was offset by 1 in some direction, so the game did not recognize it as a correct solution, as with the Triforce in figure 3. The proposed design change in this case, is to have a counter on the user interface which lets the players know how many beads are still incorrect as compared to the goal.

At the end of the interview, the developers were not only interested in adding the functionality mentioned above, but also performing a complete re-design to allow for other more complex functionality in the future, based on the discoveries made from using InVis. Some of those ideas included modeling the undesired behavior in order to automatically identify that behavior in real time in order to keep students on task; which was one of the key goals of InVis, to facilitate the development of new hypotheses by giving developer's and researchers a better understanding of their data.

In this paper we presented a visualization tool, InVis, which allows analysts to understand game-log data and the behavior of players. We were able to identify a number of interesting situations in the log-data that lend themselves to be modeled with data mining techniques in order to personalize the experiences of players of the BeadLoom Game. By modeling the player data as a behavior network and visualizing it using a hierarchical layout, we were able to display hundreds of game-log data in a concise form. The addition of case specific information, stored on edges and vertices, combined to provide a sufficient overview of how players interacted with the BeadLoom Game.

Additionally we were able to provide useful, unexpected, and surprising feedback to the game's developers which they will use in the next iteration of the BeadLoom Game's development. Based on the feedback from BLG developers it seems that information visualization has a place in game analysis and can be used to improve our understanding of player behavior in games. It is particularly useful for the generation of new hypotheses about user behavior; which when followed up with confirmation studies will aid in the research and development of video games.

6. REFERENCES

[1] E. Andersen, Y.-E. Liu, E. Apter, F. Boucher-Genesse, and Z. Popović. Gameplay analysis through state projection. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 1–8, New York, NY, USA, 2010. ACM.

[2] R. S. Baker, M. P. Habgood, S. E. Ainsworth, and A. T. Corbett. Modeling the acquisition of fluent skill in educational action games. In *Proceedings of the 11th international conference on User Modeling*, UM '07, pages 17–26, Berlin, Heidelberg, 2007. Springer-Verlag.

[3] A. Boyce and T. Barnes. Beadloom game: using game elements to increase motivation and learning. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 25–31, New York, NY, USA, 2010. ACM.

[4] R. Eglash, A. Bennett, C. O'Donnell, S. Jennings, and M. Cintorino. Culturally Situated Design Tools: Ethnocomputing from Field Site to Classroom. *American Anthropologist*, 108(2):347–362, 2006.

[5] N. Hoobler, G. Humphreys, and M. Agrawala. Visualizing competitive behaviors in multi-user virtual environments. In *VIS '04: Proceedings of the conference on Visualization '04*, pages 163–170, Washington, DC, USA, 2004. IEEE Computer Society.

[6] D. Moura, M. S. el Nasr, and C. D. Shaw. Visualizing and understanding players' behavior in video games: discovering patterns and supporting aggregation and comparison. In *Proceedings of the 2011 ACM SIGGRAPH Symposium on Video Games*, Sandbox '11, pages 11–15, New York, NY, USA, 2011. ACM.

[7] Z. Pardos and N. Heffernan. Kt-idem: Introducing item difficulty to the knowledge tracing model. In J. Konstan, R. Conejo, J. Marzo, and N. Oliver, editors, *User Modeling, Adaption and Personalization*, volume 6787 of *Lecture Notes in Computer Science*, pages 243–254. Springer Berlin Heidelberg, 2011.

[8] M. M. Rodrigo, R. S. Baker, S. D'Mello, M. C. Gonzalez, M. C. Lagud, S. A. Lim, A. F. Macapanpan, S. A. Pascua, J. Q. Santillano, J. O. Sugay, S. Tep, and N. J. Viehland. Comparing learners' affect while using an intelligent tutoring system and a simulation problem solving game. In *Proceedings of the 9th international conference on Intelligent Tutoring Systems*, ITS '08, pages 40–49, Berlin, Heidelberg, 2008. Springer-Verlag.

[9] J. C. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental evaluation of automatic hint generation for a logic tutor. In *Proceedings of the 15th international conference on Artificial intelligence in education*, AIED'11, pages 345–352, Berlin, Heidelberg, 2011. Springer-Verlag.