PROJECTIVE REPLAY ANALYSIS: A REFLECTIVE APPROACH FOR ALIGNING EDUCATIONAL GAMES TO THEIR GOALS

ERIK HARPSTEAD
Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
eharpste@cs.cmu.edu

CMU-HCII-17-107
August 2017

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

COMMITTEE:
Vincent Aleven, HCII, Carnegie Mellon University, Chair
Jodi Forlizzi, HCII, Carnegie Mellon University
Jessica Hammer, HCII / ETC, Carnegie Mellon University
Sharon Carver, Psychology, Carnegie Mellon University
Jesse Schell, ETC, Carnegie Mellon University

The research reported here was supported, in whole or in part, by the DARPA ENGAGE research program under ONR Contract Number N00014-12-C-0284 and by the Institute of Education Sciences, U.S. Department of Education, through grant R305B090023 to Carnegie Mellon University. All opinions expressed in this document are those of the author and do not necessarily reflect the position of the sponsoring agencies.

Keywords: Replay Analysis, Educational Game Design, Alignment

For my Grandfather Dale
and my Nephew Durinn

ABSTRACT

Educational games have become an established paradigm of instructional practice; however, there is still much to be learned about how to design games to be the most beneficial for learners. An important consideration when designing an educational game is whether there is good alignment between its content goals and the instructional behaviors it makes in order to reinforce those goals. Existing methods for measuring alignment are labor intensive and use complex auditing procedures, making it difficult to define and evaluate this alignment in order to guide the educational game design process. This thesis explores a way to operationalize this concept of alignment and demonstrates an analysis technique that can help educational game designers to both measure the alignment of current educational game designs and predict the alignment of prototypes of future iterations.

In my work, I explore the use of Replay Analysis, a novel technique that uses in-game replays of player sessions as a data source to support analysis. This method can be used to capture gameplay experience for the evaluation of alignment, as well as other forms of analysis. The majority of this work has been performed in the context of *RumbleBlocks*, an educational game that teaches basic structural stability and balance concepts to young children. Using Replay Analysis, I leveraged replay data during a formative evaluation of *RumbleBlocks* to highlight some misalignments the game likely possesses in how it teaches some concepts of stability to players. These results led to suggestions for several design iterations.

Through exploring these design iterations, I further demonstrate an extension of Replay Analysis called Projective Replay Analysis, which uses recorded student replay data in prototypes of new versions of a game to predict whether the new version would be an improvement. I implemented two forms of Projective Replay: Literal Projective Replay, which uses a naïve player model that replays past player actions through a new game version exactly as they were originally recorded; and Flexible Projective Replay, which augments the process with an AI player model that uses prior player actions as training data to learn to play through a new game. To assess the validity of this method of game evaluation, I performed a new replication study of the original formative evaluation to validate whether the conclusions reached through virtual methods would agree with those reached in a normal playtesting paradigm. Ultimately, my findings were that Literal Projective Replay was able to predict a new and unanticipated misalignment with the game, but Flexible Projective Replay, as currently implemented, has limitations in its ability to explore new game spaces.

This work makes contributions to the fields of human-computer interaction by exploring the benefits and limitations of different replay paradigms for the evaluation of interactive systems; learning sciences by establishing a novel operationalization of alignment for instructional moves; and educational game design by providing a model for using Projective Replay Analysis to guide the iterative development of an educational game.

ACKNOWLEDGEMENTS

A thesis is a funny thing. When you are writing one, it feels like it is the most consequential thing you have ever done—and at the time maybe it is—but at the same time there is a running joke, based on kernel of truth, that no one will ever read it. So, in a way it is like writing the most important footnote to history ever. I felt a similar way about writing my first CHI paper. All mountains seem like the tallest thing in the world until you get to the top and discover there are even bigger mountains even farther away. Adding to these complex feelings about the concept of a dissertation, is my opinion that the best research is *always* the product of collective effort. If I had it my way, single author publications would not even be allowed but that is not how this process of getting a PhD works. At the very least I want to take an opportunity to acknowledge all the other people who I feel contributed in some way to the ultimate production of this document and say thank you for helping me get here.

I wouldn't be here writing this in a very real sense if it weren't for the assistance and advocacy of a few people. My path to CMU started while I was studying abroad in Japan and figured I should have a plan for what I was going to do that summer. I looked at a backlog of opportunity emails from the psychology department of IIT and found an opportunity to be a Pittsburgh Science of Learning Center intern working on educational technology at a school called Carnegie Mellon University, which, I'll be honest, I hadn't much heard of at the time. The good news was I still had time to apply because the deadline wasn't until the next day, though being in Japan I really had 2 days. I quickly shot an email to Jo Bodnar who was the admin listed for the program and asked if I could have an extension if I could get her all of the required materials within a week and she agreed. Thanks to the heroic efforts of the IIT study abroad coordinator Louis Berends and my academic advisors at IIT Ruthana Gordon and Matt Bauer I managed to get everything in and was accepted to the program.

When I got to CMU I joined the CTAT team working mainly with Martin Van Velsen, and Jonathan Sewall under Vincent Aleven. I had had a vague idea of going to graduate school after finishing my undergrad degree but no real plan for where to go or what to study. After my internship experience I knew CMU could be a place I could thrive and, even though I didn't know much about it at the time, HCI would be the discipline I would pursue. With some guidance from Vincent, I applied to the PhD program in my senior year at IIT. The acceptance deadline came and went and I was concerned when I hadn't heard anything. I learned that due to Vincent's insistence I had been put on a wait list, which is something the department rarely does with PhD admissions. After another agonizing month of waiting I finally received the notification that a slot had opened up and I was officially accepted to the program.

In joining the HCII PhD program I have had the opportunity and privilege to meet, work with, and learn from many great people:

The fellow members of My PhD Cohort. In particular, my wife Kelly Rivers (who I met through the program) whose passion for teaching and making the world a better place has always inspired me; and whose support throughout my own dissertation work (to the point of taking dictation when I could not bring myself to write anymore) is honestly one of the reasons this document got completed in the first place. Sauvik Das, my longtime officemate and co-conspirator, who was always up for commiserating over the absurdity of academia (instead of working), and inventing myriad unrealized side-projects. Robert Xiao, the technical wizard, who was always willing (or at the very least able) to be distracted by crazy ideas and implement prototypes within hours. Samantha Finkelstein, the social adventurer, who kept all our lives interesting with her intense passion and excitement.

My longtime collaborator Christopher MacLellan who has probably shaped my research trajectory more than anyone else. After discovering a synergy between his interest in studying students learning in design tasks and

my corpus of students designing towers in *RumbleBlocks* (at a TG sponsored by Schell Games no less) we would go on to develop Conceptual Feature Extraction, TRESTLE, and the Apprentice Leaner Architecture together. For much of this work it has been hard to determine which ideas belong to who for the purposes of writing our respective dissertations. I have tried to denote some separation throughout this document but in truth I see most if not all of it as appropriately attributable to both of us.

The many advisors and mentors I have found in the faculty of CMU and beyond. Vincent Aleven, my long-term advisor, who provided both guidance and counterpoint to my research ideas, as well as the freedom to explore new threads and possibilities. Brad Myers, my early co-advisor, helped me learn how to position an HCI contribution and whose voice I still hear in my head when writing and editing papers. The core members of the PIER steering committee, David Klahr, Sharon Carver, and Ken Koedinger, who helped me to appreciate the history, practice, and theory of learning science and education. Jodi Forlizzi, who introduced me to an appreciation of the discipline of design that has become a core part of where I hope to take my research in the future. Jesse Schell and Jessica Hammer, whose insights have helped me to a deeper appreciation of game experience and game design that has become a central feature of my work. Brian Junker, Howard Seltman, and others in the Statistics Department who have contributed to my appreciation and understanding of quantitative research methods. My mentors at Microsoft Research, Thomas Zimmermann and Nachiappan Nagappan, who gave me a taste for industry research and the challenges of supporting active game development with data science.

The members of the ENGAGE Team who helped shape my early research career. The members of the ETC side of the collaboration Michael Christel, Scott Stevens, Bryan Maher and those that worked on *RumbleBlocks* Sean Brice, Matt Champer, Luke Jayapalan, Qiaosi Chen, Jing Jin, Daniel Hausmann, Nora Bastida, and Xun Zhan; as well as the other members of the HCII side of the team that I worked closely with Catherine Chase, Derek Lomas, Julia Brynn Flynn, and Amos Glenn.

The members of the CTAT and DataShop Teams, particularly Martin Van Velsen, Jonathan Sewall, Sandy Demi, Cindy Tipper, Borg Lojasiewicz, Mike Ringenberg, Octav Popescu, and Brett Leber for their endless patience during my internship and for their help and assistance during my PhD.

The incredible administrative and support staff especially Queenie Kravitz, Jo Bodnar, Audrey Russo, Gail Kusbit, and Michael Bett whose indispensable efforts ensures that make the great work at CMU possible. They helped me to appreciate that veteran admin staff are some of the most valuable members of the research enterprise.

The wonderful community of the HCII that made it more than just a research institution. The friends I made through the PIER program David and Vivi Gerritsen, Jenny Olsen, Ryan Carlson, Caitlin Tennison, and Rony Patel who functioned like a second cohort and broader support network. Dan Tasse, instigator of the HCII Album Club and fellow member (with Kelly, Robert, and myself) of the Human-Puzzle Interaction puzzle hunt team. Alex Kilbo, my life-long friend, and Mike Villena who along with Sauvik, Kelly, and I formed a semi-regular board game group that helped me broaden my game design horizons. The rotating cast of the HCII Night's Watch, who convened to share good times watching new episodes of Game of Thrones and House of Cards.

The many great PhD students past and present I had the opportunity to get to know in my time including Adrian de Freitas, Alexandra To, Amy Ogan, Anna Kasunic, Brandon Taylor, Chris Harrison, Eliane Stampfer-Weise, Erin Walker, Françeska Xhakaj, Giarad Laput, Ian Li, Iris Howley, Jason Weise, Jeff Rzeszotarski, Judeth Choi, Judith Uchidiun, Julia Schwarz, Kenneth Holstein, Kevin Huang, Martina Rau, Micahel Madaio, Nesra Yannier, Rebecca Gulotta, Stephen Oney, Steven Dang, and Yanjin Long

I also must thank my family for their love and support throughout my studies and my life; my father Stan Harpstead for fostering my scientific curiosity through many intellectual sparing matches; my mother Jodi Harpstead for engendering a sense that my work should contribute to a purpose greater than myself; and my sister Sonja Harpstead for demonstrating that following your passions whether they lead to medical school or the circus can result in great things.

Finally, as I wrote in my original application to the PSLC internship and to the PhD program "I have, through my life, been blessed with a collection of exceptional educators". It was better understanding what made these people so exceptional that drove me to an interest in research on learning and education. Honoring that purpose, I wanted to acknowledge a few of them here: John Madura, Graham Wright, Matt Bauer, Robert "Dr. Bob" Schleser, Mattox Beckmann, Connie Aiken, Dan Butler, and Leah Higginbotham.

TABLE OF CONTENTS

List of Figures

LIST OF TABLES

List of Equations

List of Abbreviations

CHAPTER 1 INTRODUCTION

Ever since the first settlers struck out on the *Oregon Trail* in 1974, educational games[1] have been a part of the instructional landscape. Of course games and play have a much more storied history in the cultures of the world [68] and games of one kind or another have been used for educational purposes for centuries [143]. I begin with *Oregon Trail* not because it was the first educational game but because it has some personal significance to me[2] and because it is one of the most immediately recognizable examples of a modern digital educational game. Though it may not seem like it when compared to recent examples like *Foldit* [38] or *Crystal Island* [129], the game instantiates a highly detailed model based on historical data [125] and gives players the chance to explore the consequences of their actions while remaining faithful to the experiences that real settlers had on the trail. Giving players the ability to interact with detailed models of concepts and understand the implications of their actions are some of the potential benefits of using games in educational settings [52,152].

Despite the modern academic study of educational games having been around for some time [52], there continues to be active debate in the literature as to whether games can be good for learning. Every few years a new meta-review is published drumming up the question again [33,37,45,54,105,171]. While these reviews may be useful for policy makers or curriculum developers to decide when to begin adding games to a curriculum, I believe they are asking the wrong—or at least an unhelpful—question with regard to games and learning. Asking if a game*,* difficult as that term itself is to define [135], could be beneficial for learning recalls the debates between Clark and Kozma on whether the medium of instruction would ever influence learning over or beyond the method of instruction used [34,81]. At best this question is already solved, requiring only an existence proof, of which there are several [20,42,56]. A far more useful question, in my opinion, is this: how can we design better educational games?

I am not the first person to ask this question. There are several frameworks on educational game design that seek to inform this question [8,13,16,52,168]. However, a majority of these frameworks falter in that they describe an ideal to aspire to in design without providing much guidance on the process of how to get there. For example, James Gee's influential account of *What Videogames Have to Teach us About Learning and Literacy* [52] contains a list of principles that describe the qualities of good entertainment games that could leveraged to an educational purpose. Similarly, Amory [13] and Annetta's [16] frameworks describe a number of facets that quality educational games possess and make arguments about how those components could contribute to learning when done well. Aleven [8] and Winn's [168] frameworks take an approach of describing different perspectives to look at educational game design, similar in spirit to Schell's lenses for general game design [135], which provide a means of framing educational game design from the perspective of different intellectual traditions. What most of this work seeks to do is provide a common language with which to talk about the challenges and opportunities of educational game design. This is a noble goal as a lack of common terminology is a problem game design generally struggles with [39]. Where I take issue with this work is that none of these frameworks, to my knowledge, have been demonstrated to support an active design process, focusing instead on applications of critique and post-hoc analysis. This is like telling a settler that they could have a better life in Oregon when all they really want

[1] Throughout this thesis I will refer to "educational games" as a category mainly because my work is influenced by my participation in the Program for Interdisciplinary Education Research (PIER) at CMU and thus involved games in K-12 settings teaching conventionally academic subjects. I would expect many of my methods and findings to be applicable to other formulations of "serious" or "transformational" games despite my word choice here.

[2] I went to elementary school blocks from the headquarters of the Minnesota Educational Computing Consortium, developers of Oregon Trail.

to know if which direction is West. Rather than develop yet another framework of what a good educational game is, I instead want to focus on how better to steer designs toward the desired outcome of a good educational game. Exploring this line of inquiry requires some understanding not only of what a good educational game should be, but also why designing such a game is hard.

For some readers, it might be sufficient to point to the nature of design as a wicked problem [24] in justifying why this task is so hard; however, the educational game design task has unique challenges of its own. Even when armed with the most recent findings of learning science theory, there are several trillion ways one could design an instructional intervention [76]. In addition to the inherent complexity in instructional design, games, particularly the kind of open-ended simulation games commonly used in educational settings [152], employ a delicate balance of mechanical systems that can result in emergent behavior [133]. This property of games can make it difficult to anticipate how a design choice will affect the behavior of the system and subsequently the learning of players. While domain theory and subject matter experts may provide some insight, judging the result of a design alteration is ultimately an empirical question.

Further complicating the problem of educational game design is the fact that a player's experience in a game is not solely the product of a game's mechanical systems. Hunicke, LeBlanc, and Zubek provide a framing for this issue in their Mechanics, Dynamics, Aesthetics (MDA) framework [69], where they describe the aesthetic experience of the player arising from the player's interactions with the mechanics of the system, which they call dynamics. In this framing, designers and players have fundamentally different perspectives on a game. While designers see more of the mechanics and rules of the game they create, players perceive a dynamic aesthetic experience while the game is being played. Others have referred to this relationship as a second-order design problem:

> *"As a game designer, you are tackling a second-order design problem. The goal of successful game design is meaningful play, but play is something that emerges from the functioning of the rules. As a game designer, you can never directly design play. You can only design the rules that give rise to it. Game designers create experience, but only indirectly." - Salen and Zimmerman* [133].

A similar sentiment is echoed in the learning science literature to describe the challenge of instructional design. Herbert Simon, one of the founding fathers of the fields of cognitive psychology and artificial intelligence is noted as having said:

> *"Learning results from what the student does and thinks and only what the student does and thinks. The teacher can advance learning only by influencing what the student does to learn." – Herbert Simon*[3]

A key to both of these perspectives is that the agency over the experience lies in the hands of the player or learner. Put another way, educational game designers cannot directly cause learning to happen; they can only ensure that whatever players do in their games affords the ability to learn desired concepts.

[3] I have endeavored to find a citable version of this quote in print and have come up empty handed. Simon is attributed with having said a version of this phrase often in the opening of How Learning Works [12] and Carnegie Mellon University saw fit to set the words in stone for a memorial to him, so I believe I am justified in quoting it here.

From this lens, I view educational game design less like planning a clear instructional path and more like tending a garden of possible learner experiences. The challenge in this approach is understanding the space of possible experiences players might explore, characterizing whether those experiences are upholding the goals of the design, and, in the cases where they are not, taking steps to trim or guide the experience to align it with the original intention. This orientation has more in common with the reflective [136] and unselfconscious [9] processes of traditional design practices rather than the front-loaded goal-directed processes of conventional instructional design [101,166].

The perspective of Donald Schön is one that I find particularly useful in this context. He describes the general design process as a reflective conversation with the materials of a design situation [136] and further characterizes that conversation as iterating through stages of seeing-moving-seeing [138]. Initially designers see the context of their design situation and perceive some problem to be solved. They then make a move to correct that problem by altering the situation in some way. Finally, they can see both the intended and perhaps unintended consequences arising from their move and the cycle starts anew.

Schön's perspective on the design process relates to the current question of educational game design in that the types of seeing that Schön talks about are intrinsically difficult for designers to do in an educational game setting. The influences of players' agency and their dynamic interactions with a game's systems [69] require some representation of the player experience to be present in order to see whether a game is meeting its goals. Further, the learning events that are the goal of an educational design take place in a player's mind and are invisible to designers [77]. Therefore, an open area of research is developing better methods for designers to see their games in a way that lets them make new moves and consider the consequences of those moves.

Literature on the game design process as a practice provides some insights into addressing these challenges. Many game design thinkers highlight the importance of establishing experience goals to serve as a guiding focus throughout the design process. Schell talks about starting by establishing what the designers want the essential experience of a game to be [135]. Fullerton advocates for taking a play-centric design approach where it is the designer's role within a team to be an advocate for the player and their experience [51]. Culyba's forthcoming framework on transformational game design [40] advocates for establishing transformation goals early as a way to build a common language with stakeholders. The Tandem Transformational Game Design approach of To *et al.* [154] is a unique example of not only advocating for the establishment of transformational player goals at the beginning of the process but also acknowledging that the goals themselves can be subject to iterative development as a design evolves. These processes have much in common with instructional design practices for developing alignment with curricular goals [166].

From a process standpoint existing methods for game design emphasize an iterative approach that can be responsive to change, often invoking Agile software development practices [22]. Development is commonly separated into phases of conceptualization, pre-production, production, and refinement [51]. In the early stages designers focus more on what the core on the experience is using low-fidelity prototyping and other brainstorming methods [30]. It is these early stages where Vandenberghe advocates for failing fast and following the fun [158], to more quickly hone in on a solid idea to pursue further. In the production stage, prototyping becomes more formal but still iterative. The RITE method [98], for example, advocates for a process of rapid prototyping and quick iterative testing of potential design solutions as soon as they are feasible to implement. The final stages of design are focused on polishing and tuning the player experience to ensure it is meeting the desired goals.

Running through the entire process of game design is the common practice of playtesting. Because of the emergent and dynamic nature of player experience [69] it is impossible to evaluate the quality of a game without some representation of the player. Playtesting gives designers the ability to see the experiences their games create in action and evaluate whether they are meeting expectations. In their study of playtesting Choi *et al.* highlight that it is important that playtesting sessions be organized in a purposeful way [30] whether it be to explore the space of player interest, or prove to a stakeholder that a game is having it desired impact. Further, Schell [135] argues that playtesting should be viewed apart from other kinds of evaluation such as usability evaluation or software integrity testing, as the kind of testing of unique interest to game designers and for game design problems.

While playtesting is a powerful method for understanding game design issues there exists an opportunity to expand upon it. For example, Open-ended simulation games allow for many possible player experiences and it can be hard to see and appreciate them all in the context of a single playtest. Further, educational games are often intended for and tested in classroom settings, adding to the complexity of collecting useful insights from observation. These issues can be addressed by augmenting the practice of playtesting with the common game development idiom of replay systems [44]. Designers can utilize detailed recordings of playtesting sessions and modern game analytics techniques [89,140] in order to better understand the experiences players are having and whether those experiences are upholding their intentions.

Broadly, this thesis explores the question of how we can use novel replay-based analytics techniques to support the design process to make better educational games. This broad question includes two particular thrusts.

The first thrust is concerned with answering the question of how to define *better* educational games. For this thrust, I turn to the idea that all educational games are being designed with the goal of teaching something to their players. Thus, the quality of an educational game is measured in how well it embodies its target content and conveys its target concepts. In instructional design, this relationship is called alignment. Existing approaches for measuring alignment are ill-suited for the space of educational game design, so I have developed a novel analytical technique for characterizing alignment in educational games.

The second thrust addresses the replay-based analytics approach itself and explores how it can be used to support the iterative design process. As part of this thrust, I have developed the Replay Analysis approach that encompasses a collection of methods for using replays to inform the game design process. Within this broader category, I explore two forms of Replay Analysis. Retrospective Replay Analysis uses historical recordings of game play to provide data to game analytics techniques in evaluating the current state of a design. Projective Replay Analysis augments this process with an AI agent, which enables designers to explore next iterations of a design without having to gather new playtesting data.

In my earlier work, I employed Retrospective Replay Analysis to explore ways of measuring this idea of alignment within the educational game *RumbleBlocks*. I then developed Projective Replay Analysis to address the challenge of predicting the effect new design decisions have in the context of new play traces based on old replays. The driving research question behind this work was: could a method like Replay Analysis give a reliable picture of what players would do in new situations, and could the insights gleaned from these approaches provide useful direction for the design of a game.

Throughout this document, I detail my explorations of the concepts of alignment and Replay Analysis in the design of educational games and describe the contributions they make to the fields of educational game design, learning science, and human-computer interaction (HCI). This document will be structured in the following way:

Chapter 2 opens by discussing the concept of alignment in instructional design and its theoretical underpinnings. I then describe several existing methods for measuring and using alignment in the design process and how these methods fall short in the context of educational game design. Finally, I define my own characterization of alignment and how I operationalize it for measurement.

Chapter 3 introduces Replay Analysis as an approach for educational game analytics. I start by describing related literature on game analytics, the use of replay based approaches in game and user interface evaluation, and the application of AI models to the evaluation of games and interfaces. I then define Replay Analysis itself and describe a taxonomy of replay forms including Retrospective Replay, which uses historical data to understand the current state of a design, and Projective Replay, which takes historical replay data and projects it into a future iteration of a game to test whether it has improved. Further, Projective Replay can be implemented as either Literal Projective Replay, which instantiates a naïve player model that tries old actions in new context without reasoning, or Flexible Replay, which augments prior log traces with an intelligent AI model that makes decisions based on new information. Finally, in this chapter, I describe the implementation of the Replay Analysis toolkit as a concrete software tool and systems contribution of this thesis.

Chapter 4 will describe the assumptions I make about contexts that this work is applicable to, specifically single-player step-based puzzle games. Additionally, this chapter will describe the game *RumbleBlocks,* an educational game design to teach structural stability and balance concepts to children in K-3 (ages 5-8). *RumbleBlocks* could be seen as a prototypical example of the type of games that the techniques I describe in this thesis would be appropriate for. Further, *RumbleBlocks* will be the specific context used in the remaining chapters to demonstrate the utility and validity of replay analytics to improve alignment.

Chapter 5 presents a formative evaluation of *RumbleBlocks* that served as the first application of Replay Analysis to understanding an educational game. I explore initial questions of whether players are learning target concepts from playing the game and whether the game is aligned to its instructional goals. This work provided evidence to validate my conceptualization of alignment by showing a correspondence between concepts with good and bad alignment and players' learning.

Chapter 6 demonstrates the capacity of Replay Analysis to allow designers to see their game from several angles in diagnosing what problems it may have. This chapter elaborates on the work of the formative evaluation by exploring the solution space of *RumbleBlocks* with the goal of explaining why the patterns of misalignment seen in the formative evaluation may have occurred. I ultimately find a pattern of behavior in the results that suggests a possible cause of misalignment related to particular sub-structural faults in players' solutions rather than broader patterns related to the game's goals.

Chapter 7 explores the power of Projective Replay Analysis to enable designers to see the implications of next-iteration design choices without gathering new data. This work takes the form of a replication study of the formative evaluation of *RumbleBlocks* and serves as a validation of Projective Replay Analysis. Building on the findings of Chapter 6, I developed several variations of *RumbleBlocks* with the intent of improving alignment. These variations were tested using Projective Replay Analysis and one was tested with a new classroom study. Comparing the results of these studies, I find that Literal Projective Replay predicted a change in alignment but Flexible Projective Replay did not. Further, producing a new game variation based on alignment analysis did not produce a better alignment when tested.

Chapter 8 concludes the document with a discussion of the results and what they imply about Replay Analysis and alignment. Ultimately, I find that Projective Replay Analysis has promise as a method for understanding next

step design variations, but more work is needed to improve its ability to function on more divergent designs, particularly when Flexible Replay is used. Results for alignment show similar promise for use of the concept as a tool to inform design, but further study will be needed to better understand the relationship between alignment and student learning gains. Finally, I close with a discussion of the implications this work has on future investigations.

CHAPTER 2 ALIGNMENT

At the heart of the iterative design process are a pair of simple questions: "Is my product good?" and "If I were to make a change, would it be better?". These questions are a characterization of what Alexander calls fitness in design [9]. They are also the questions at the heart of what Schön describes as move-testing hypotheses [136] that designers make while they work to bring a particular design into line with its goals. In order to develop better methods for aiding educational game designers, we must understand how fitness is operationalized within the context of educational games.

Within the broader context of instructional design, the idea of the fitness of a design for its instructional purpose has commonly been called alignment. In many discussions, alignment is used to frame other issues, such as content/construct validity in assessment [85] or implementation fidelity of a new intervention [66], but it can also be considered an instructional design principle in its own right. In addition to being a principle, there is also evidence that aligned instruction leads to better learning. Cohen details several early studies of the effect of alignment on learning and generally finds that aligned instruction often leads to 4-to-1 effect sizes over unaligned instruction [36]. Within the context of educational game design, Habgood and Ainsworth [56] have explored the related concept of intrinsic vs extrinsic integration (i.e., how directly learning content is integrated into the core interactions of a game). They found that players of an intrinsically integrated math game learned more than players in an extrinsic condition and were willing to continue playing the game for longer when given a choice.

In the literature, alignment research has commonly included one of two areas: curriculum alignment or instructional alignment. The focus of curriculum alignment is to demonstrate a correspondence between the content of a curriculum plan and a set of standards, often associated with questions of material coverage [15]. Instructional alignment, on the other hand, addresses the relationship of instructional goals, instructional moves, and assessments, often within the context of a single classroom [36]. One could look at curriculum alignment as being related to outward accountability of instructional practice, while instructional alignment is more focused on an internal validity. For the purposes of this thesis, I am more interested in the space of instructional alignment.

As the name would suggest, alignment normally refers to an agreement between two or more components of the instructional context. However, what those components are varies depending on the definition being used and the purpose of the analysis. I refer to this issue as instructional alignment's n-body problem. In physics, the n-body problem refers to trying to predict the individual motions of n bodies, accounting for all of their individual gravitational interactions on each other. The problem is easy for two bodies but gets significantly harder, if not impossible, as more bodies are added to the system. Similarly, instructional alignment is almost always discussed as a relationship between two things (e.g., state assessments and their related standards [96]), but there are many such binary relationships in a larger educational system. For example, one might look at how well a teacher's instructional actions relate to the particular curriculum they are required to teach as a measure of accountability [121]. Alternatively, one could look at how the content of an assessment relates to a set of standards, at a state or inter-state level [123]. Just as in physics based n-body problems, measuring alignment requires the assumption of a common reference frame.

In most discussions of alignment, the common reference frame is provided by a focus on three main components of the instructional context: goals, instruction, and assessments. This trio of elements appears in many existing studies of alignment [36,41,96,123] and will serve as a useful language for discussing alignment across studies going forward. While these three components establish the main elements of the instructional context that should agree with each other, they leave open to interpretation what agreement means (i.e., how alignment is measured), and further, how instruction can be designed towards alignment.

Designing for Alignment

One of the main uses of alignment in prior work is as a frame around the design process. In these examples, alignment takes on a methodological bent, providing guidance on how the design process should proceed with the intent of arriving at a good instructional design.

In his discussions of instructional alignment, Cohen [36] describes criterion-referenced instruction as an older method of instructional design drawing on the behaviorist tradition of B.F. Skinner. The general pattern of criterion-referenced instruction is that instruction and assessment are done with identical tasks; in effect, literally teaching to a test. The result of this process is to merge assessment and instruction into one entity, and to define goals as accomplishing the assessment. This approach makes less sense in traditional educational settings but has applicability in domains such as training, where the goal is not to increase individuals' transferable skills or knowledge, but to reduce variance across performers of a common task.

Evidence-centered design [101] is a method created by the Educational Testing Service for developing assessments in novel contexts. At a high level, the process requires defining a series of models: a student model, which encompasses the collection of goals that the instructor wants students to learn; a task model, which describes the structure of the task the student is performing; and an evidence model, which describes a mapping between tasks and the competencies they evidence. While this process is primarily associated with assessment design rather than instruction, it has shown promise towards integrating assessment into instruction (e.g., in stealth assessment [144]) and has previously been used in educational game design settings [130]. Within the context of the trifecta of alignment, evidence-centered design has an outside-in approach, where it begins with a set of goals and a set of tasks (i.e., instruction) and attempts to define a mapping between the two with an evidence model (i.e., assessment).

Finally, in backwards design [166], the instructor begins by defining the goals of the instruction, usually relying on standards or other curricular targets, then defines assessments that could be used to measure whether students have achieved the goals, and finally plans a program of instruction that will lead students to improve performance on those assessments. This approach is called backwards design to counter the initial intuition of many teachers to begin designing with the means of instruction (e.g., favorite materials or lessons), and instead focus on the ends of instruction (i.e., the goals). From the perspective of alignment, this approach defines goals to be the foundation upon which everything is based, then builds toward instruction in a linear fashion.

Each of these methods can be valuable scaffolds for the design process and provide a useful framing for reasoning through issues, but there are limitations to relying on methodology to produce good alignment. While it is not unreasonable to assume that good products will result from good practice, having simply followed a particular method does not provide a sense for how well aligned a current design is without some form of objective measurement. The educational game design process can be complicated by issues like expert blind spot [78], the complexity of ill-defined domains [90], and the wickedness of design problems in general [24] and each of these could contribute to the introduction of alignment problems by accident. These limitations echo HCI literature on design process where there is a distinction between designing the right thing and designing a thing right [155]. In so far as the structure of alignment implied by each of these methods is appropriate to a context, they can be helpful frameworks for designing toward alignment. However, they still have the issue of leaving the concept of agreement between elements open to interpretation.

Measuring Alignment

A number of techniques exist within the literature that either explicitly claim to measure alignment or have implications for the task of measuring alignment. I summarize a few of the key approaches that have implications for the development of a general method for measuring instructional alignment in educational games.

In thinking about how to quantify the alignment of a design, one might be tempted to use a pre-posttest measurement as evidence. A typical paradigm of assessment driven instructional design is to define some form of pre-posttest that is administered before and after an instructional intervention. If an improvement on this test is observed, the usual conclusion is that the instruction was aligned to the assessment because learning was observed. While this is a gold standard scientific design, I would argue it is not informative but rather confirmatory. If the result is negative or inconclusive, then little can be concluded about what aspect of the instructional behavior within a system broke down. A method that is inconclusive in the negative is a problem for iterative design because, through the course of iteration, a designer is likely to deal with far more bad prototypes than good ones [53]. In measuring alignment as a means of guiding the design process, a more nuanced formulation is necessary.

A series of methods have been developed as part of accountability and educational reform movements [41] that I collectively refer to as matrix methods. A good summary of some of the principal examples of this type can be found in Marone and Sireci's review [96]. The usual procedure of these methods is to define some kind of matrix that compares instructional units, assessment items, or standards to some level of competency (commonly a level of cognitive depth or a series of levels similar to Bloom's taxonomy of learning objectives [82]). The general idea of this approach is that a set of standards would mandate that a student possess some knowledge or skill to certain level of depth, so a lesson or assessment should cover that content to the required level. Once the matrix is defined, a group of subject matter experts are convened as a panel of raters. The raters then rigorously annotate the target standards, instruction, or assessment items by marking appropriate matrix cells. In some cases, a cell is filled in with the mere presence of a competency-by-level match, and sometimes it is scaled based on amount of overall instructional time reserved for the concept. Ultimately, the result of the rating process is a matrix for each instructional component being compared. The relationship between a pair of matrices can be quantified in a number of ways [96] based on how well the patterns of cells overlap. Other quantifying methods have also been explored that result in heatmap-like visualizations for inspection beyond a single summary metric [122].

While the matrix methods are commonly considered to be a gold standard within their particular tradition of literature [120], they do have some issues that would make them difficult to apply in the context of iterative educational game design. For one, the process of convening a panel of raters is extremely time intensive and expensive. The most common use case for methods of this kind is in the evaluation of large scale reform efforts where the costs of detailed review are more justifiable. In a tighter loop of educational game design iteration, having subject matter experts review all content changes would likely become infeasible quickly. A second issue with many of these methods is that they tend to assume the evaluation of static material. In this way, they are effectively solving the first-order design problem in that they look to evaluate whether the intended content is present in the material without regard to how a learner might experience it. As I previously discussed, the experience of playing a game is a dynamic one and is shaped partially by the input of players [69]. This quality would require that raters be able to evaluate all possible experiences, or provide ratings based on their own play experiences, which would run the risk of expert blind-spots [78].

A third approach that I see as relevant to alignment measurement, though it has not to my knowledge been billed as such in the past, is the practice of Knowledge Component model refinement embodied by the PSLC Datashop [75,153]. A Knowledge Component (KC) is a formalism from the Knowledge-Learning-Instruction (KLI) framework

and represents "a specific unit of cognitive function that is inferable from performance on a set of related tasks" [77]. One of the theoretical aspects of a KC is a useful stand-in for other constructs like skill, fact, ontology, production rule, schema, etc. The process of KC model refinement starts by first defining a mapping between KCs and tasks, often called a Q-matrix [21]. Student performance data is then captured from the use of a system and fit using a specialized statistical model called the Additive Factors Model (AFM) [28] to generate learning curves plotting the change in students' performance over time under some assumptions of learning. By inspecting these curves, designers can see where their expectations about learning are being met and where they are not [153]. Anomalies in the learning curve highlight places where a learning task does not embody the content designers thought it did and can lead to improvements in instructional design [79]. I have applied this KC model refinement approach to the context of educational game design in the past and found benefits for its use in considering alignment [58].

While a KC modeling approach to alignment has promise, it too has some limitations to its application in an educational game design context. One of the biggest points against the KC modeling approach is the necessity of a very fine-grained description of problem solving. The method was originally developed in the context of intelligent tutoring systems that operate under a paradigm of display-based reasoning [84] where discrete unit steps in problem solving are given immediate feedback [159]. Not all games can support this kind of structure. Further, developing a reasonable KC model can be a time intensive task requiring detailed cognitive task analysis [5], often with the assistance of domain experts.

Operationalizing Alignment

Educational game designers need a formulation of alignment that allows them to consider how well the instructional behavior of their game aligns to their instructional goals in a way that is tractable within an educational game design context. In order to account for the complexities of the second order nature of game design, we need an informative operationalization of alignment that can do more than confirm whether or not a game is working. Such an operationalization needs a way of relating a game's instructional moves (i.e., feedback) to an assessment of player understanding within the context of the game itself. Drilling analysis down to a finer grain size of instructional moves allows for a more nuanced evaluation of alignment issues. To this end, I define alignment with a paradigm of internal assessment linked to instructional feedback moves that the game makes in response to player actions.

The essential logic behind my approach to alignment is that a game's feedback and incentive structures are some of the main ways a game communicates meaning to players. In so far as the pedagogical intent of an educational game is for players to come away with a better understanding of the content the game is meant to be about then it is crucial that these systems behave appropriately. I do not intend to claim that this is the only way to evaluate whether a game is upholding its intended meaning, as players can derive meaning from many aspects of game experience [133], including re-appropriating a game in transgressive play [1] or creating meaning around it in a social context [72], but remaining grounded in feedback—a well-regarded instructional design principle [14] in its own right—provides a concrete basis on which to build an evaluation technique.

Understanding the concepts of feedback moves and player actions requires a brief discussion about the idea of game spaces. All games take place in some kind of space. One way of thinking about this space is as the Magic Circle [133] that forms a boundary around the game and gives its components meaning (e.g., outside the Magic Circle, *Monopoly* money is just paper). However, game spaces have other useful properties beyond their ability to give endogenous value to game elements. Another useful lens for talking about game space is in terms of *functional* game space [135]. The functional space of a game can be conceptualized as the space in which the

game *really* takes place. For example, while the game of *Monopoly* is printed as a two-dimensional board, in terms of functional space, the game is really a single one-dimensional loop of properties. Further, each property is actually a zero-dimensional space, as the particular placement of a player piece within the bounds of a property is meaningless.

Using a lens of functional space allows us to define a player action as anything that meaningfully changes the state within the functional space of a game. Extending this concept further, a solution to a game challenge or level is a collection of player actions that lead to a goal state. This perspective allows us to define the solution space to a challenge or level as the set of all pathways through, or configurations of, functional game space that lead to goal states. Borrowing from the Knowledge-Learning-Instruction framework [77], I make the assumption that, in creating their solution to an in-game challenge, a player is expressing their understanding of the knowledge or skills (i.e., KCs) required to solve that challenge. This perspective is also similar to what Plass called assessment mechanics, which provide players with the tools to be able to express their understanding of concepts [119]. Under the assumption that a player's knowledge is expressed through their play, there would arise a qualitative (and possibly quantifiable) difference between solutions created by someone who understands a concept and someone who does not. Markers of this difference can be formalized as a separation function of the solution space in terms of target domain principles.

Feedback can be defined in similar terms as the collection of changes to the functional game space that the game communicates to players in response to their actions. While this collection of changes is potentially very large, it can be useful to think about it in terms of different channels [135] that are functionally distinct. For example, successfully solving an in-game puzzle might result in a bright green check mark, animated fireworks, and a "woo-hoo" sound; all of these responses are multiple channels of conveying the same message, namely, that the player succeeded. Similar to how players with different understandings will generate different portions of a solution space, feedback can act as a separation function over the solution space, where a certain collection of solutions will receive one type of feedback while the rest will receive another.

Using this operationalization, solutions within a solution space can arrive at one of four designations, best thought of as the 2x2 matrix shown in Figure 1. Two quadrants in this matrix are desirable and, if solutions consistently land in either one of these quadrants, it would indicate that the game is well aligned. Solutions that are highly principled would ideally be given some form of positive feedback, which would imply that the game is reinforcing target concepts to the player. Similarly, solutions that are unprincipled should be given negative feedback, which would mean that the game is discouraging deviations from target concepts, allowing a player to learn from their mistakes. Solutions would ideally *not* fall into the other two quadrants, where principled solutions are discouraged or unprincipled solutions are reinforced. In these cases, the game is sending contradictory feedback to students, at best confusing them and at worst fostering misconceptions.

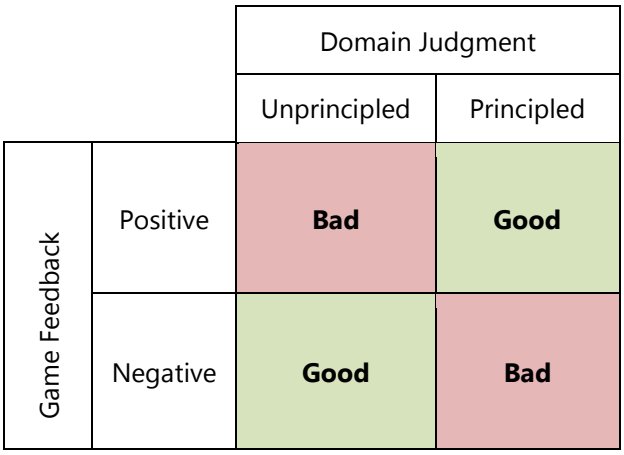


Figure 1. A matrix showing the possible alignment interpretations of student solutions based on the agreement between domain judgment and game feedback.

Ultimately, I define the alignment of an educational game as the level of agreement between the game's separation function of its solution space (i.e., feedback), and a domain-principle-based separation function of that same solution space. I formalize the measurement of agreement as the coefficients of a regression model that predicts game feedback using metrics tied to domain principles. I refer to these metrics as Principle-Relevant Metrics (PRMs) in that they are metrics that are intrinsically tied to a principle a game is trying to teach.

The formalism of PRMs does impose a constraint on the method in that I assume some kind of proxy metric for adherence to a principle is calculable from a given player solution; however, the definition of these metrics can be quite broad as the framework of regression modeling is flexible. For example, a metric could be a calculable number from a game state, or it could be a categorical tagging based on some set of rules, which could then be treated as levels of a factor in regression. The heart of the approach is to ensure that relationships that should hold true within the domain also hold true in the game's behavior.

Viewing alignment as agreement between how feedback separates a solution space and how assessment separates the same space, it would seem like the obvious design solution is to base feedback directly on an in-game assessment; however, this is not always possible. For example, the target domain might be ill-defined [90], meaning that there is no single strong domain theory that could be used to create feedback rules, or any such rules would be subject to debate by experts. Further, it could be that the principles of interest have complex interconnected effects on the domain such that representing them faithfully requires tuning and balance. Balancing complex systems is a common task in game design [70], but it often takes many hours of testing to make minute changes and is rarely straightforward [55]. Finally, while it may be possible to based game behavior directly on a principle based model it may be prohibitively expensive to implement a sufficiently accurate model for the task. In such cases simply black box solutions, such as an off the shelf physics engine, might be more feasible but would require monitoring to ensure instructional goals are still being upheld by the system's behavior.

In all of these cases, playtesting and iteration are essential to see if the instructional behaviors of a game are aligning with designers' expectations. Since the instructional behaviors of a game encompass the results of many different design decisions, we need a definition of alignment that is capable of functioning at a fine grain size. I believe my definition of alignment as an agreement between separations of a solution space serves this purpose.

CHAPTER 3 REPLAY ANALYSIS

The second order nature of game design [133] makes it difficult for designers to see and evaluate a design without some representation of the player experience. I present a potential solution to this problem in the common game idiom of in-game replays [44] and developed the technique of Replay Analysis to explore such an approach.

Replay Analysis is the core methodological contribution of this thesis. At a high-level, Replay Analysis is a game analytics technique that uses in-game replays of player sessions as a data source to support evaluations of game design. In addition to supporting evaluation of a current game design, the approach can also make use of a computational theory of human learning to instantiate AI player models to project replays into future versions of a game that players have not yet seen.

In this chapter, I will describe the broader space of prior work that Replay Analysis contributes to before laying out the Replay Analysis approach as I define it in my work. Finally, I will detail the implementation of the Replay Analysis toolkit I developed for facilitating Replay Analysis in the Unity game engine.

Game Analytics

The measurement of player experience has been a growing topic of interest for both general game user research and serious games research [89,140]. Many practitioners and researchers have explored different ways of measuring player experience [107] including self-report and subjective surveys [23,113], biometrics and physical response data [100,108], and data mining and analytics [46,74,157]. Among these approaches, I believe the data mining and analytic approaches are the most promising for guiding alignment analysis in terms of solution space.

Game analytics research is generally concerned with the application of game log data (also commonly referred to as telemetry) to answer questions about game players and game design. There are many approaches to game analytics and several ways to describe the relationships between different methods (see [140] for several different taxonomies). In moving toward a design informative process using analytics, it is necessary to understand where analytics sits within the iterative design process. Schön describes the design process as a reflective conversation with a situation [136]. This conversation iterates through stages of reframing a design situation, moving to improve the situation, and then evaluating whether the move resulted in improvement. Commonly, analytic approaches occupy the evaluation stage of this loop, as they provide a picture of a current design in terms of a specific framing of the design problem. Existing approaches also support reframing to varying extents in their ability to ask questions beyond their original intent. What is generally missing, however, is the ability to explore move testing. Rather than employing analytics as a tool throughout the conversation with a situation, a designer must step away from their context and ask for it to be analyzed before being able to form new design hypotheses to test.

Using Schön's notion of a reflective conversation as a guide, I categorize different game analytics techniques into one of three groups based on two distinctions. The first distinction I make is whether an approach is generally measure-then-record or record-then-measure.

Measure-then-record approaches to game analytics work by performing the actual measurement of a desirable feature within the game and then recording only the result of that measurement. This tactic is generally the realm of very large-scale metric-based [46] approaches to understanding player experience where key performance indicators that designers want to record are known well in advance [80]. The benefit of these approaches is that they generally require minimal post-processing of data beyond simple aggregation or statistical testing, because the desirable information exists directly in game logs. However, a measure-then-record approach has limited capacity to inform reframing of a game design situation, because what can be said about the player experience

is only what can be inferred from the measurements that exist in the data, as any other context was lost. Because of their capacity for scale and general rigidity, measure-then-record approaches are usually employed for later stage monitoring of games post release.

Record-then-measure techniques take the approach of recording some kind of representation of the player experience from which measures are distilled. This style of game analytics is far more common in educational game work with several existing examples [117,128,145]. The trace-based recording approach common to intelligent tutoring systems is also an example of this kind of analytics [19,75]. The general benefit of these kinds of approaches is that they preserve some portion of the context of the play experience, allowing for that experience to be reframed to some extent. The second distinction in my categorization of prior analytics approaches is within the group of record-then-measure approaches and asks whether an approach is designed to record-to-measure or record-to-capture.

Record-to-measure paradigms are typified by a focus on recording player experience in service of a future measurement that is planned. The quintessential example of this kind of approach is Evidence-Centered Design [101,102,130]. Though not strictly an analytics approach, Evidence-Centered Design is focused on an extensive process of building evidentiary arguments about learner competencies before any measurement takes place. Record-to-measure techniques are generally more appropriate for making the kinds of evidentiary arguments needed during the *prove* stage of playtesting [30], where the focus is on persuading stakeholders that a design is working rather than informing design refinement. While these approaches have strong validity for developing assessments, I would argue they are less suited for informing iterative design because they are generally committed to the lens of their assessment, which can limit the ability to reframe the context beyond that measurement.

Record-to-capture, on the other hand, generally intends to capture the player experience as it happened and defers any measurement or characterization of the experience until after it has been captured. This is commonly where extremely high fidelity and labor-intensive techniques, like video recording, are employed to provide ground truth for other methods [114,127]. These forms of analytics are desirable early in the refinement stages of playtesting [30] because they are the closest to the traditional form of design evaluation via observation. However, the maintenance of so much context runs the risk of drowning in excessive detail, leading context-heavy analytics approaches to be discouraged by the broader community of research [11,47,88].

Prior Replay Based Approaches

Within the broader spaces of Game Analytics, Learning Analytics, and HCI Usability methods, there are a number of techniques that have used replay of various forms to understand designs and systems. To better characterize how my approach to Replay Analysis differs from this prior work, I will review some of the major examples and general approaches that exist in the literature.

The use of replay to evaluate user experience has existed as a concept in the HCI literature for some time. As far back as 1984, Neal and Simons described a system called Playback [111] that recorded a users' key strokes on a terminal based system that could then be played back through a prototype interface. This system was primarily interested in supporting measurements of command use frequency and timing information. Given its age, the design of Playback is limited to very specific hardware platforms, but it represents an early example of what would now be called a click-stream, or raw input, recording of user interaction [151]. In these approaches, raw input signals are fed back into a system to reproduce user behavior. A similar technique is used in the modern practice of tool-assisted speed running (http://tasvideos.org/) where players record their raw controller inputs to be played

back for verification in competitions or performance at events [99]. These raw input based replay approaches are often limited to systems with discrete control systems (e.g., pressing keys or a limited number of buttons on a controller as opposed to the analog input of something like a joy stick or dragging a mouse).

In the intelligent tutoring systems community replays of various kinds have been used to understand and improve learners' experiences. Baker, Corbett, and Wagner describe levels of replay fidelity for analyzing learner behavior [18]. The goal of their analysis was to understand forms of replay that could allow human observers to watch learners' experiences and apply observational coding techniques [114] that could be compared to data-mining analyses of log data. In their taxonomy, a high-fidelity replay would be a video recording or exact visual reproduction (what they call full screen replays) of a learners' interactions with a tutoring system, while a super-fidelity replay would augment that video with additional information such as eye-tracking or fMRI data. The main focus of their work, however, was exploring what could be done with low-fidelity replay in providing simple textual descriptions of sessions. In their study, low-fidelity text replays were generated by randomly sampling a starter learner action from a session and then adding the following actions up to a window of 20 seconds and presenting these actions as a text transcript. They found that, for their purposes, observers using textual replays were able to reach a sufficient level of interrater reliability in coding for behaviors like gaming the system. While these results are interesting, they are more useful for facilitating human interpretation of replayed events for validation of automated methods rather than directly supporting analytics themselves.

Another use of a replay-like system from model-tracing intelligent tutoring [3] is what Aleven *et al.* called "meta-cognitive model tracing after the fact" [7]. Normally, within model-tracing intelligent tutoring, learners' actions are traced against a model of expert performance to measure their current ability in terms of the expert model. In developing a new model of help seeking Aleven *et al.* used log traces of learners to see how often their preliminary models detected certain kinds of help-seeking behavior and whether those designations corresponded to student learning as theory would suggest. This usage of replay is geared more toward theoretical ends rather than design as the target of replay and iteration is the cognitive model, not the student's experience, though the resulting model could be used to drive the feedback systems of a tutor in the future.

The field of software engineering has also used replay based approaches for automated testing and debugging [106]. A particular example of this approach is Dolos by Burg *et al.* which provides low overhead deterministic replays of web interactions to support debugging Javascript systems [26]. One of the core foci of Dolos is to detect execution divergence and inform developers about unexpected behavior. The intention of software engineering replay systems is the exact reproduction of a program's execution state at a memory level in service of replicating execution errors and software bugs. This level of detail is excessive for most evaluations of design related issues, and implementing systems capable of guaranteeing exact reproduction of program state to a memory level requires costly focused engineering efforts.

The use of replay log files is also common in the space of game analytics for various purposes such as strategy analysis [162] and prediction [164]. This work differs from my own goals here in that it is directed toward understanding the player decision making process rather than evaluating the quality of given game. Replays also feature prominently in the space of game AI, particularly for strategy games like *StarCraft* [165]. In these use cases, the replay log files are usually not being employed for their intended purposes (i.e., reenacting a game or match), but are instead being used as a means to get around having to instrument telemetry hooks in a commercial game.

Prior Uses of Player, Learner, and User Models in Evaluation

A subcomponent of my replay approach deals with the question of integrating AI models[4] of players. A number of other researchers have employed modeling techniques for various purposes in the design of games, instructional environments, and user interfaces generally. In this section, I will describe several domain specific taxonomies of these approaches and detail some of the notable examples.

In addition to writing about design as a reflective conversation with the material of a situation, Schön also describes four different purposes that AI researchers could take in applying AI to the context of the design task [137], which I find useful to frame prior work. In the first level of his taxonomy, Schön suggests that researchers could develop an AI that achieves a similar design output to a human but not necessarily in the same way a human would (similar to the Turing Test [156]). The second level is to develop an AI that reproduces not only equivalent output but also reproduces *how* people go about designing. The third purpose is to develop AIs purely as assistants to designers while the fourth purpose is to use AI approaches to develop environments aimed at better understanding the design process as a target of research itself. Within my work, I am primarily interested in the third level of AI as design assistant, but there are several interesting implications of the other levels that play into this purpose.

Within the domain of intelligent tutoring systems, VanLehn *et al.* present a similar taxonomy of how AI models of simulated students can be used for educational purposes [161]. The first use case is for teachers to practice teaching with simulated students to formally verify teaching effectiveness. The second use case is to have students learn alongside simulated learners to benefit from collaborative learning effects. This form of simulated learner has been demonstrated in the work on SimStudent [86,97], where students tutored simulated learners to explore the benefits of learning-by-teaching. Finally, the third use case is for instructional developers to use simulated students to test their products before release. This final case is the primary focus of my own work. Employing simulated students as a means of playtesting can enable designers to see into the experiences of their games using a tighter iteration loop than one which requires new playtesters. In terms of Schön's framework, I see this application of AI primarily as a third category application of AI in design, but there are some implications for his other categories. For example, should such testing agents only be required to perform adequately in an environment (Schön's category 1), which is likely easier to develop, or should they also be designed to perform in a humanlike way (Schön's category 2). The development of the Apprentice Learner Architecture [94], which I contributed to and use in my work, was partly directed at this distinction. In an evaluation colleagues and I were able to show an Apprentice Learner Model applied to an intelligent tutoring system could not only reach similar asymptotic performance to human learners but also follow a similar learning trajectory, suggesting a commitment to Schön's second category is possible.

HCI also has a broad literature on the use of user models in the evaluation of interfaces. This literature mostly derives from the work on GOMS models [71], which is itself based on early theoretical work on human problem solving and information processing [27,112]. In a GOMS analysis a researcher must specify the Goals, and sub-goals, of a task being performed; Operators, which are primitive actions that can be taken in service of a goal; Methods, which are sequences of operators that accomplish goals; and Selection rules, which determine which methods and operators to apply in the case that multiple could be applicable to a given situation. The goal of a GOMS analysis is to generate timing estimates for using an interface, accounting for various assumptions about human cognition and performance. Each primitive operator is assigned a timing weight that is used to score an

[4] Within this context of replay, when I refer to a "model" I mean a runnable computational model of behavior rather than a statistical model that simply predicts numerical outcomes.

agent's execution based on the operators it applied. One of the key distinctions in this work from my own focus is that a GOMS analysis usually assumes it is modeling expert performance, whereas in an educational context, I am more interested in modeling novice performance and ideally the change in performance as a learner progresses from novice to expert.

Covering the entire space of applications of AI to games is well beyond the scope of this thesis [170] but I will explore a few cases that I find relevant to Schön's third category of AI in design. One of the largest uses of AI in game design is within the space of Procedural Content Generation (PCG) [73]. An example of AI use in a PCG context is generating levels with guaranteed desirable qualities [142,147]. Often these approaches function within the context of a formal description of game mechanics like the Video Game Description Language (VGDL) [134], which is powerful in that it allows the field of AI research in games to advance under a common understanding, but also potentially limiting as it imposes a representational formalism on designers.

AI in games has also been used to explore automated playtesting of game designs [109,150]. This work is similar in spirit to VanLehn's description of simulated learners being used to test instructional software, but has a key difference. Similar to my issue with GOMS modeling, prior uses of AI in playtesting games tend to be focused on expert performance. There is a noted counterexample by Holmgård *et al.* [67] who explored the use of intentionally hamstrung AI models as a proxy for modeling different skill levels. This work is encouraging but does not make use of findings from the cognitive sciences in a way that I feel is necessary for educational game design.

Replay Analysis

Within the context of my work and this thesis, I define Replay Analysis as **the use of in-game replays as a data source to support varied analysis**. The core intuition behind this approach is that a running game engine has access to any potentially relevant pieces of game state in service of analysis, while a simpler log trace is restricted to what information explicitly exists in, or could be inferred from, the logs. A clear distinction that I want to make upfront is that these replays are not video based, but rather action traces of players' session that can be recreated within a game engine. Further, the analyses being performed do not need to involve any visual representation of the game play at all, though they certainly can. The main requirement for replay is that the underlying model of the game can be re-instantiated in the same state it was in during the recorded play session.

One of the biggest strengths of Replay Analysis generally is that it affords designers and researchers the ability to decouple the process of defining metrics for evaluating a game and the designing of the game itself. Since analysis is performed over replays, telemetry hooks for particular metrics of interest do not need to be fully decided on before playtesting can begin. I do not necessarily advocate for a fully decoupled process of game design and evaluation design, as it is often a good idea to be clear on what will be measured before design begins [101,166], but delaying a hard focus on proving a game's effectiveness can allow for more exploratory playtesting [30] to refine experiences and try new directions. Also, in a research context, Replay Analysis enables researchers to use old data for new analyses (i.e., secondary analysis) to validate and improve upon prior work. Further, even for analysis within a given set of research questions, replay can be useful, as it can be hard to foresee what analytics or extracted variables would be necessary prior to collecting data.

Forms of Replay Analysis

Within my formulation, Replay Analysis can take several forms (Figure 2) depending on how it is performed. Each form uses replay traces in different ways and each enables different kinds of analysis. The first form of Replay Analysis is what I call Retrospective Replay Analysis. In retrospective replay, recorded sessions are played back through the game they were recorded from and used to better understand the current state of that game.

Retrospective replay has been used in most of my prior published work [58,60,62][5] to enable several kinds of analysis of educational games (see Chapter 5 and Chapter 6 for more details on this work). The core focus of a retrospective Replay Analysis is to create a faithful reproduction of the game as the player experienced it and then interpret that reproduction with metrics or abstractions as suits analysis.

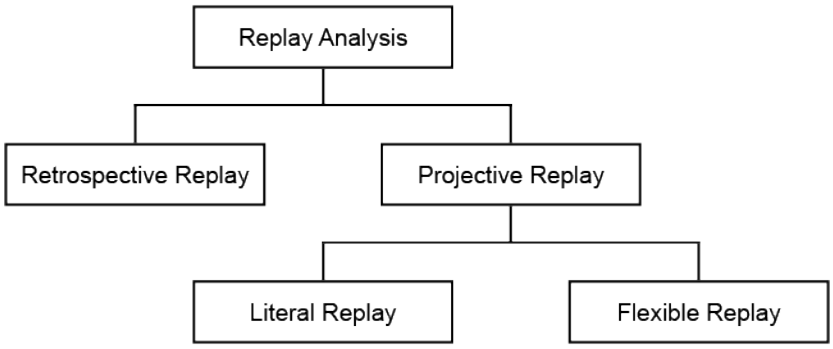


Figure 2. The taxonomy of Replay Analysis approaches showing how different forms relate to each other.

The second major form of Replay Analysis is Projective Replay Analysis. In a Projective Replay Analysis paradigm, recordings of player sessions are used as input to a computational agent, which interacts with the game as if it were a player and generates a new trace that is interpreted for analysis. The main benefit that Projective Replay adds over retrospective replay is the ability to run analyses on a next iteration of a game without recruiting a new population of playtesters. At the simplest level, Projective Replay could be used for tasks like regression testing [106]; but in more complex situations, it can be used to provide insight into how a new yet similar set of players might interact with a game after some change was made. An example case and validation study of Projective Replay is show in Chapter 7

Projective Replay is further divided into two forms depending on how the player agents are structured. In Literal Replay the original action traces are enacted exactly as recorded but within the new game environment. This form could be seen as a naïve player model that only attempts actions it has tried in the past without learning from the consequences. In cases where prior actions are no longer possible, that agent can simply move on or report a failure. At first glance Literal Projective Replay may seem similar to Retrospective Replay, but there are some differences, particularly in where the authority of the game resides. In Retrospective Replay, the trace takes precedence over the entire game, overriding or ignoring in-game triggers and events; whereas, in Literal Projective Replay, the agent is subject to the rules of the game as executed. For example, if a game mechanic were changed such that some player attempts reach a goal state earlier than they originally did, Retrospective Replay would continue to apply actions to the level, while Literal Projective Replay would advance to the next level as directed by the game system.

The other sub-form of Projective Replay is Flexible Replay Analysis. In Flexible Replay, the player model is augmented with an AI decision making process. In principle, the AI decision making process could be implemented in a number of ways; however, I advocate for a player model that adheres to assumptions about human learning by taking demonstrations from original players' replays and learning to perform its own actions within the game. It is important that the computational player models are informed by prior player recordings

[5] Note that in my prior work Retrospective Replay Analysis was referred to simply as Replay Analysis. With the introduction of Projective Replay Analysis in this thesis I find the extra distinction useful to reduce confusion.

instead of rules or training from the game creators to avoid the possibility of expert blind spot [78]. One of the purposes behind user testing is to learn something about a product that was not already known, and training AIs with current designer understanding runs the risk of reflecting the designers' existing conceptions back at them.

It is certainly possible that other sub-forms of Projective Replay could be envisioned. For example, an Exhaustive Projective Replay agent could explore the space of all possible approaches to a game level without input from existing log data. This is the approach taken by several existing AI-aided game design strategies [148–150].

Replay Analysis Toolkit

In order to facilitate the process of Replay Analysis within my own research, I have developed the Replay Analysis toolkit, which I have presented previously [62] and include here as a systems contribution for this thesis. The system is built as a C# library to integrate with the Unity game engine, but the high-level architecture of its design should be generalizable to other languages and game engines. The reference implementation of this system can be found at (https://github.com/eharpste/ReplayAnalysisEngine). The full system has two main pieces, a logging system and a replaying system, but I will describe the system in three sections because the addition of the agent system to support Flexible Projective Replay adds an extra layer of complexity.

Replay Logging Library

The first component of the Replay Analysis Toolkit is a logging library for recording actions during a play session. The purpose of the logging library is to provide a simple API for game developers to record players' actions in a way that can facilitate replay at analysis time.

The recording of actions follows in the educational data mining tradition of the PSLC DataShop [75]. The action encoding format used by DataShop is based on the plug-in architecture for tutoring [126]. In this system, Ritter and Koedinger described an architectural design that could be implemented into any existing piece of software to communicate with an intelligent tutoring backend. The goal in my case is similar, to augment any game implementation with the capacity to communicate with a replayer, but there are some differences.

The largest difference between my logging approach and the one used by DataShop is that the DataShop specification captures transactions as interactions between students and intelligent tutoring systems by pairing a student tool action with a tutor response, while my approach only records the student's action. In designing the system this way, I make the assumption that the game engine will have the capacity to reconstruct the "tutor response" while replaying, and thus recording the explicit response is unnecessary. Further, in the Projective Replay case the explicit response may no longer be valid.

Within the logs, players' behavior is captured at the level of a basic action which I define as the smallest unit of meaningful action that a player can exert on the functional game space. This grain size is similar to the notion of a semantic event in the plug-in tutor architecture [126] and is analogous to what Schell calls operative actions [135], which are the base actions a player can take within a game. These actions are contextualized to the game world, (i.e., picking up or dropping an object) rather than the raw input of the player (i.e., mouse down at position (x_1, y_1), mouse up at position (x_2, y_2)). I contextualize actions rather than raw input events because actions that are semantic to a given game design are more robust to change over iterations in design than the meaning of a given set of screen coordinates, which can be dependent on hardware and platform configurations.

The log specification of a single action is based on the DataShop specification of a Selection-Action-Input (SAI) triple but augments it with a further State field, resulting in the 4-tuple of Selection-Action-Input-State (SAIS). For

a given action, the Selection is defined as the entity[6] the player is acting on, in most cases the name of the entity, or some other unique identifier. The Action is what the player is doing to the entity, effectively the name of the verb being done. The Input is a set of parameters to the action being performed. The meaning of the Input field is contextual to the type of action, and may not always be meaningful, such as in the case of button presses that use "-1" for Input as a standard convention. An alternative way to think about the relationship of the elements of an SAI is as a semantic description of an underlying function call[7] where Selection is the object being called, Action is the function itself, and the Inputs are the parameters to the function. The State property of an action is a description of the relevant game state at the time the action was taken. The State property is used during the replay process to allow for arbitrary indexing into a session without having to interpolate the game state in order to recover context. The paired recording of state is also important in situations where a game's state and behavior could change for reasons other than direct player action (e.g., a physics engine simulating the motion of objects, or a non-player character making its own independent decisions). The emphasis on contextualized action paired with state is meant to embody a record-to-capture paradigm of analytics and maximize the amount of information to be available for future reframing and move testing.

In addition to the SAIS properties, actions are also recorded with a series of unique identifiers to enable easier aggregation. At the top level, all actions are associated with a given User ID. Every time a player starts a new game they are assigned a unique Session ID to be associated with that session of play. A session is composed of a series of attempts, each with an associated Attempt ID, on the levels of the game, which include a Level field for identification. Finally, each action is tagged with a unique Transaction ID and Timestamp at time of action. This aggregation structure does impose an assumption that the game is composed of a series of levels that are attempted until the player is successful or not, but it is a common game structure that is applicable a large number of educational games [116] and is similar to the format of most DataShop logs [75].

The logging library was written in C# and is designed to be straightforward to use. For each player action, the implementer adds a call to a function that takes as parameters the Selection, Action, and Input to record the action. For convenience, there are also simpler versions of this call that record the standard properties that all Unity GameObjects have, such as Transform properties for position and rotation. The recording of state is facilitated through a system I built into the library that is similar to, but distinct from, Unity's tagging functionality, where GameObjects that would be relevant to record are aggregated by user defined tags. By default, GameObjects' names and transform properties are captured in the state description, but a *LoggableObject* script can be extended to augment state information with further data as needed for a particular game.

There are several other similar log recording systems for Unity including that ADAGE system from Owens and colleagues [116], as well as Unity's own analytics service (https://unity3d.com/unity/features/analytics)[8]. The novelty in my approach is in the commitment to the SAIS structure for recording player actions and in having the ultimate goal of supporting further analysis through replay (recording-to-capture) rather than being a general-

[6] The use of the term entity throughout this section is in reference to the software pattern of an entity-component model, of which Unity is an example. See (http://entity-systems.wikidot.com/) for more information.

[7] This perspective derives from the history of the plug-in tutor architecture, whose action descriptions were based on the structure of Apple Events in Apple Script that described semantic actions through functions for the purposes of creating macro action recordings.

[8] My implementation, in fact, predates the release of Unity's own analytics product.

purpose analytics suite. In so far as those purposes could be supported by another logging solution than my approach could be transfer.

The Replay Analysis Engine

The system for replaying player log traces is called the Replay Analysis Engine (RAE). The RAE reconstructs players' sessions action-by-action, provides software hooks for running analyses during replay, and integrates with an AI framework to instantiate player models based on prior logs for testing new game mechanics. It is the ability to augment a replay with different analysis scripts that sets my replayer apart from prior approaches to in-game replay [44]. In this way, the live game engine is used intrinsically as the data source for analysis rather than inspecting textual traces of logs, as is more common in educational data mining and other analytics work [146].

The several classes and components[9] of the RAE are designed to be analogous to many of Unity's normal components. The core processes of the replayer are factored across four main components: a central *ReplayAnalysisEngine*, which controls the main flow of replay and delegates operations to the other components; an *Agent* component, which provides actions to be performed given the current state of the game; an *Interpreter*, which performs interpretations of the game state to produce analysis; and a *GameSpecificReplayComponent (GSRC)*, which acts as a bridge between the general replay process and the specific logic and processes of the current game being analyzed.

In general, the main replay loop is similar to Unity's update loop, but implemented as a series of co-routines to allow the two to execute in parallel. A single tick of the replay loop starts by requesting a new action from the *Agent*. Once an action is obtained the *GSRC,* it is used to assume the state described by the action and then performs the action itself. The normal game logic is then allowed to run until a defined stopping condition is met (e.g., waiting for some amount of time, or waiting for all physics objects to stop moving). At this point, the main game logic is paused and the *Interpreter* is executed to produce some interpretation of the action and its effects. The *Agent* also evaluates the result of the action to support learning processes. The *ReplayAnalysisEngine* then proceeds to the next iteration and the cycle repeats.

The *Agent's* main task is providing actions to be performed, but how those actions are created differs depending on the type of replay being performed. In a Retrospective or Literal Projective replay, the *Agent* serves actions out of a database or log file in time order, grouped by user, session, level, and attempt. The difference between the two is that in Retrospective replay the agent continues to feed actions as they exist in the log without regard for the game's own logic; whereas, in Literal Projective replay, the agent can be interrupted by the game's logic as it advances to new levels or resets levels as needed. Using a series of messaging functions, the *Agent* notifies the *ReplayAnalysisEngine*, and by extension all other core components, when a new action would be from a new attempt, level, or user to trigger behavior like reloading the current level or resetting the entire game. In a Flexible Projective replay, the *Agent* produces actions to perform based on AI reasoning and learns from the results of those actions as the game progresses. I will describe this learning process in more detail in the next section.

The *Interpreter* component provides little outward functionality itself other than a single function used to trigger an interpretation. The simplicity of this specification is designed to enable a wide range of possible analyses. While running an interpretation of a particular player action, the *Interpreter* can leverage the full affordances of Unity's APIs to generate any metrics, feature abstractions, or descriptions necessary for a given analytical goal.

[9] As with entity, the term component has particular meaning within the context of the unity engine. See (https://docs.unity3d.com/Manual/UsingComponents.html) for more information.

The *GSRC* performs several tasks throughout the replay process. Its main tasks are to parse the SAIS data from a player action, instantiate the state, and perform the action. While the inclusion of this component does impose a burden on the game developer, they will have already written code to record the action in the first place and could, in many cases, simply direct the relevant SAI information to existing logic in the game's codebase. In addition to enacting player actions, the *GSRC* is used by the *Agent* during Flexible Projective replay to generate descriptions of the current game state and actions performed by the *Agent* for use in training during the action evaluation stage. This description process is similar to describing actions for the purposes of logging and could in many cases use the same logic.

In adapting the system to their own game, a developer will generally only need to implement their own *GSRC* and *Interpreter* with generic versions of the other components provided by the library. The core components of the engine (with the exception of the *ReplayAnalysisEngine* itself) inherit from a common *RAEComponent* base class that provides several messaging functions (similar to existing Unity functions like OnCollisionEnter or OnLevelWasLoaded) to allow housekeeping logic to be spread throughout the main replay process. Additional subclasses of this common base class could be used by a developer to inject additional special logic into the core replay process as needed.

In addition to the core *RAEComponent*s, the system also uses a *ReplayBehavior* component that is attached to all existing GameObjects in a scene by the RAE. The main purpose behind this system is to provide the RAE with a method for interacting with the various GameObjects in the scene without relying on Unity's normal infrastructure, in the event that it is being used to perform the game's own logic. For example, the *ReplayBehavior* component implements the same system for tagging GameObjects that is used by the logger in order to designate objects that appear in state messages or as targets of agent actions. The *ReplayBehavior* component also provides various helper functions similar to those provided by the *LoggableObject* class during play time.

Flexible Projective Replay Agent

In both Retrospective and Literal Projective Replay, the *Agent* component makes use of a log database to feed actions directly from prior players' traces into RAE. In Flexible Projective Replay, this process is replaced by an AI agent system that takes prior players' traces as training data and generates new actions through its own reasoning.

The current implementation of the Flexible Projective Replay Agent is inspired by the Apprentice Learning Architecture [94]. Colleagues and I developed the Apprentice Learning Architecture as a computational theory of human learning from demonstrations and feedback. The goal of the architecture is to try to explain human learning by describing its mechanisms rather than merely modeling behavioral patterns with statistical techniques. The overarching theory generates computational models of human learning (i.e., runnable programs that instantiate assumptions of learning) that can be executed to generate behavior. This behavior can then be compared with existing empirical results at a fine grain level to evaluate whether the models and the theory they are based on are suitable explanations of the learning process.

For the purpose of this thesis, I am not interested in the use of the Apprentice Learning Architecture for advancing theories of human learning[10]. Instead, I am interested in leveraging the Apprentice Learning Architecture's goal of modeling a human-like learning process to model the behavior of novice learners in a game setting. This sets

[10] For this use case of the Apprentice Learning Architecture see Christopher MacLellan's thesis [92].

my use of agents apart from prior explorations of virtual playtesting by maintained a strong commitment to cognitively informed modeling rather than creating models merely capable of playing a game.

The apprentice learner model used in Flexible Projective Replay learns a collection of skills that it can execute in the game world from example player actions and associated feedback. These skills take a similar form to production rules, IF-THEN rules that generate a particular behavior (THEN) when their conditions (IF) are met. The model factors skill learning into three sub processes: how, where, and when. For a given SAIS description of a player action, *how learning* seeks to explain how the player came up with the specific set of inputs given information in the state and a set of primitive operators; *where learning* seeks to generate a pattern that describes where the elements used in the how explanation came from; and *when learning* uses the state and game feedback to understand the conditions for when this particular skill should be applicable in the future.

For *how learning,* the model uses a forward-chaining theorem proving algorithm [131]. For each of the inputs in the SAIS, the algorithm takes all of the facts in the state and a set of primitive operators (e.g., functions for doing basic arithmetic, or for noticing when two values are equal), and generates a space of possible explanations by successively applying the operators to the state until a specified depth of explanation is reached. The system then randomly samples among the set of the shortest explanations that produce the target input value. The intuition of this choice is that a shorter explanation is likely to be better. If no valid explanation can be found, the system uses a simplistic explanation by saying that the exact values from the example did not derive from features of the state and much therefore be special constants.

Where learning uses a specific-to-general relational learning algorithm [103]. This process takes the variables referenced by the operators in the explanations generated by the how search and tries to construct a pattern of conditions that describes where those variables originate in the structure of the state. Initially, the conditions are all the relations in the state that refer to the specific variables of the first example. As more examples for the same skill are added, the algorithm generalizes the pattern by introducing new variables for constants that change across examples and dropping relations that are not common to all examples. At any given point, the generated pattern is the most specific description that is consistent with all the previously seen examples for a particular skill.

Finally, *when learning* makes use of the TRESTLE algorithm [93] (see Appendix B for more details on TRESTLE) to learn a classifier for when the skill is applicable. To train the classifier, the state of the SAIS is paired with the feedback from the game and incorporated into the skill's TRESTLE knowledge base. The result of this process is something akin to a probabilistic decision tree for each skill that can return whether the skill should fire given any structured state.

Initially, the agent starts with no skills. On the first demonstration the agent receives, it creates a new skill using *how learning* and then updates the skill's *where learning* and *when learning*. On subsequent demonstrations, the agent first checks to see if any of its existing skills could have generated the values in the SAIS. If there are any matches, each of the matching skills' *where learning* and *when learning* components are updated and no *how learning* is performed. If no skills matched, a skill is created with *how learning* and updated with *where learning* and *when learning*.

The training process for the agent alternates between two primary orientations: watching and trying. Initially, the agent begins in the watching orientation as it possesses no knowledge of its own with which to play the game. When watching, the agent returns actions from the log database as it would in Literal Projective Replay. As the agent plays, actions it performs are accumulated in a list for the current attempt until the end of the level when feedback is provided. This feedback is then associated with each of the actions and the agent learns from them

in the order they were executed. Once the agent has learned from all examples, it switches into the trying orientation.

When in a trying orientation, the agent is provided a representation of the current state of the game and asked to generate an action to apply. The agent iterates over its current list of learned skills, testing the state against the *where learning* pattern and *when learning* classifier of each skill. If the test returns true, a new SAIS is created by applying the skill's *how learning* explanation to the current state to generate a grounded action. This action is then returned to the RAE to be executed in the game. If the agent cannot produce an action on its own, it requests a new demonstration action from the log database and tries to generate an action again with the result of the demonstrated action.

Throughout the interactive learning process, there are several situations where the process could break down that I call impasse points. The most prominent impasse point is when an agent has executed several actions under its own reasoning but then hits a point where it does not know what to do. In this case, the agent is several actions ahead of the log trace and requesting a demonstration may not be applicable to the current situation in the game. When this impasse happens, the agent reverts to the point where its reasoning diverged from the log and picks up where it left off, watching demonstrated actions. In the current implementation, the reasoned actions that led to the dead end are forgotten.

Another impasse point is when an agent's actions result in no perceptible change in the game state. If a single action results in no change to the world, then the agent immediately trains that action as a negative example and tries again. This process encourages the agent not to try actions that do nothing. A more nuanced version of this impasse, however, is when an agent's actions result in a recurring cycle of game states. In these cases, the agent would run indefinitely without intervention. In the current implementation, I address this issue by keeping a list of previously seen game states for the current attempt. If the agent ever detects such a cycle, it drops the entire attempt and moves on to the next level.

CHAPTER 4 CONTEXT OF APPLICATION

My main goal in this thesis is to demonstrate the capacity of Replay Analysis to facilitate alignment-informed iteration of educational game design; however, the space of all possible educational game designs is enormous and not even the best-intentioned method could hope to cover all of it. In this chapter, I will lay out several assumptions I make in my Replay Analysis methods that provide bounds around the types of games that my methods could be applicable to. Further this chapter will also describe the educational game *RumbleBlocks*, which will serve as both a prototypical example of the kinds of games that my methods could be applied to as well as the main context for the work described in the chapters that follow.

One of the foremost assumptions I make in Replay Analysis is that a game is single-player. This orientation is not intended to be a fundamental commitment that single-player games are superior or desirable for education but rather derives as a consequence from the various intellectual traditions on which my work is based. For example the Apprentice Learner Architecture [94] and the Knowledge-Learning-Instruction framework [77] that provide much of the cognitive basis for my work have primarily been applied to single learner paradigms in the past, though there are some noted recent examples to expand into collaborative contexts [115]. Limiting to single-player games also serves as a simplifying assumption to avoid having to model complex multi-agent systems during replay. It would not be impossible to expand the replay paradigm to a multi-player context as the mechanisms that are used to create replay mechanics often resemble the same software patterns that enable networked multiplayer games [44], however, for the time being such efforts are reserved for future work.

A second major structural assumption I make is that a game is step-based in its interactions. By this I mean players act upon the environment by taking a series of sequential steps to arrive as some solution to an in-game challenge or whole level. This paradigm could be seen as opposed to continuous or real-time play that might be seen in a first-person shooter game such as *Quake*. Again, this assumption derives from the historical approach of intelligent tutoring that informs much of my work, however, what constitutes a step can be more broadly defined than it may appear. In his discussion of various representational approaches for steps in tutoring systems VanLehn arrived at a definition that a step is "the smallest possible correct entry that a student can make" [160]. Combing this perspective with Schell's notion of functional game space [135] one could view a step as the smallest unit of player action which causes a change to the functional space of a game. While it may seem obvious that a game like *Dragon Box* is step-based, by virtue of performing discrete manipulations of cards in the environment, one could also see a game like *Portal* as being step-based problem solving where discrete units of meaningful interaction drive puzzle solving forward. Linehan *et al.* demonstrated an orientation similar to this perspective in their exploration of the challenge curves[11] of four successful puzzle games [87].

I am hesitant to invoke a particular game genre as the focus of my work as the notion of game genres is fraught by a tension between industry established categories and other more aesthetically-based topologies [17]. One of the more promising taxonomies for game genre is Heintz and Law's Game Genre Map [65]. Within their taxonomy, the puzzle genre fits most closely with the assumptions of game structure built into my work, however, I could also see potential application to adventure, simulation, or strategy genres as well. Ultimately, the context that my work focuses on could best be described as single-player step-based puzzle games.

[11] Linehan *et al.* refer to these as learning curves but I would frame them more as curves of challenge introduction, hence my change in terminology. I discussed this somewhat in my own learning curve paper [58].

RumbleBlocks

Much of the work contributing to this thesis has been in the service of better understanding the game *RumbleBlocks* [32], which could be seen as a prototypical example of the kind of game my work is relevant to. Not only have these explorations aided the understanding and design of this particular game but they also serve as an extended case study of how Replay Analysis can support design iteration of an educational game by enabling a number of different analytics methods. To be clear, I do not intend *RumbleBlocks* to be taken as a contribution of this thesis itself; however, the game is the context in which much of the thesis work was done. In order to fully appreciate the implications of the balance of this document, it is necessary to first have some context on *RumbleBlocks* itself. Throughout the rest of this chapter, I will detail the original design of *RumbleBlocks* and its educational goals. In the next several chapters, I will then proceed through the stages of an extended case study of *RumbleBlocks* and demonstrate how Replay Analysis aided the exploration and iteration of the game.

RumbleBlocks (Figure 3) is an educational game designed to teach basic structural stability and balance concepts to children in kindergarten through grade 3 (5-8 years old). It was originally developed as part of the DARPA ENGAGE program at Carnegie Mellon University, which was a collaboration between game design students at the Entertainment Technology Center and learning science researchers in the Human-Computer Interaction Institute. The ENGAGE project had a broad mission to explore the design of games to support young children learning core STEM content, scientific inquiry, and social and emotional concepts [4]. For the purpose of this thesis, I will only be reviewing *RumbleBlocks'* effectiveness as it relates to the core STEM content of structural stability. More information about the design history of the game and a playable version can be found at: http://www.etc.cmu.edu/engage/?page_id=840[12]

RumbleBlocks was originally designed to focus on three core principles of structural stability:

1. An object with a **wider base** is more stable.
2. An object with a **lower center of mass** is more stable.
3. An object that is **symmetrical** is more stable.

These principles are derived from goals outlined in the National Research Council's Framework for New Science Educational Standards [110] and other science education curricula for the target age group.

Figure 3. A screenshot of *RumbleBlocks*.

[12] For archival purposes, I will try to maintain a copy of a playable version of *RumbleBlocks* at: www.erikharpstead.net/rumbleblocks

The game follows a science fiction narrative wherein a group of aliens have their mothership damaged by a comet and must escape to a nearby planet. Players must help the stranded aliens by constructing towers out of a set of provided blocks[13] that capture energy dots floating in the scene to power the ship and allow the alien to fly away. Each level starts with the player finding an alien stranded on a cliff and a deactivated spaceship left on the side of the scene (see Figure 3). The player's goal is to build a tower out of blocks that is tall enough to reach the alien so that they can give the alien's ship back by placing it in a glowing target zone. In the process, they must also capture all of the energy dots in the scene, or else the ship cannot launch. Once the player has placed the ship on top of the tower and captured all of the energy dots, the ship powers up, which triggers an earthquake. If the spaceship falls out of its target zone, either by the tower collapsing or the ship falling off the tower, then the player fails and must restart the level; however, if the tower remains standing with the ship on top, the player succeeds and progresses to the next level.

Each set of levels in *RumbleBlocks* is designed to focus on a different principle of stability. The targeting of different principles is accomplished mainly through level design (i.e., the mechanics do not change depending on the target principle; only the positions of energy dots and types of blocks available change). The energy dots can be used to both scaffold and constrain students' solutions to a level, forcing them to prioritize one principle over another. However, even with this scaffolded design, there are an unknown number of possible valid solutions to any given level, because the earthquake mechanic relies on the dynamics of Unity's real-time physics engine to evaluate the student's structure. That is, even though the level designer may intend for a particular tower design to be the solution, other designs may also work.

The unknown nature of a given level's solution space entertains the possibility that players could create solutions that ignore the target principle for the level and still succeed. Such a situation would allow learners to complete the game without having to contend with the entire set of target principles. While one might expect that the physics engine takes care of most of these misalignment cases in *RumbleBlocks* (i.e., regardless of how well a solution embodies the target principle for a level, it is still subject to the laws of physics), there are many design decisions that can affect this behavior. For example, changing the mass and friction properties of blocks can alter how likely a structure is to collapse during an earthquake. Additionally, altering the speed or magnitude of the earthquake can also affect what happens to a player's solution and thus whether the core concepts of the game are being properly represented. This complication of a dynamic feedback mechanism presents a challenge in anticipating whether the game is providing properly aligned feedback across the myriad of possible player experiences.

[13] During one of the playtests, a player suggested that the blocks were parts of the broken mother ship, which had never occurred to me, but it makes a nice narrative sense.

CHAPTER 5 FORMATIVE EVALUATION OF RUMBLEBLOCKS

In order to demonstrate the Replay Analysis approach and my formulation of alignment for this thesis, I will describe a number of studies. This first study is part of a formative evaluation of *RumbleBlocks*, and for the goals of this thesis, it serves multiple purposes. On a basic level, this work represents the first demonstration of the method of Replay Analysis and the RAE in an educational game. Further, this work was the origin of my formulation of alignment as a regression of principle-relevant metrics to game feedback and provides some encouraging evidence for that approach. Finally, in the broader context of my work, the data collected during this study is used throughout all of my future investigations as a baseline for comparison in iteration. The work in this chapter has previously been published in [62], which presented the original design of the Replay Analysis toolkit, and [60], which expanded further on the concepts of Replay Analysis and its utility.

At the culmination of *RumbleBlocks'* original development cycle, the ENGAGE team was interested in evaluating whether the game was succeeding at its pedagogical goals. As part of this development effort, I assisted with a formative evaluation of the game. The original goals of the study were two-fold. First, this study was a first formal evaluation of *RumbleBlocks'* instructional effectiveness generally; second, it was a study to explore the effect of a series of levels based on a contrasting case instructional paradigm on students' learning. I will only focus on the general formal evaluation question, as the results of the contrasting case comparison have been published elsewhere [29]. The formative evaluation was done as a series of in-class playtests paired with out of game transfer tests that took place over four sessions: an external pretest, two 40-minute sessions of play, and an external posttest. This work was performed in two Pittsburgh area public schools with 281 students participating across both schools. For replay based analyses, the full sample of 281 students is used; however, only 174 students were present for all four days of the study and have full data for the pre-posttest analysis. (See Appendix A for further descriptive statistics from this study.)

To facilitate the evaluation, two sets of levels were selected to be used as in-game pre- and posttests, counterbalanced across players. These levels (one for each principle) were chosen out of the normal pool of levels but were altered to remove the energy dot mechanic and to prevent players from retrying after a failed attempt. These special levels were placed after a short collection of tutorial levels, which explained the basic mechanics of the game, and at the end of the game. On the second day of the study, all players were jumped to the second set of pre-post levels to ensure they all got a chance to play them. This design allowed us to get a sense of how players built before and after they had experience with the game. In addition to the in-game evaluations, players also took out-of-game paper and pencil tests, before and after playing the game. These tests contained items relating to stability and construction, based on the three principles of base width, low center of mass, and symmetry.

Replaying for Pre-Posttest Measures

As a first pass at evaluating the effectiveness of the game, I leveraged the pre-posttest design to confirm whether *RumbleBlocks* was succeeding in getting players to improve in their understanding of its target principles. As I stated previously, such an analysis would primarily be confirmatory with regard to alignment evaluation, but it is nonetheless a valuable analysis tool when developing an educational game. If positive learning results were found, it would be reasonable to assume that the game was well aligned; but if results are not positive or conclusive, then the interpretation with regard to alignment becomes unclear.

The results from the formative evaluation study were promising. The out-of-game tests showed a slight, yet significant increase in player's performance from pretest to posttest, using a paired-samples t-test, $t(173) = -2.13$, $p = 0.03$, $d = 0.16$. Looking at players' pass rates on the in-game pre-post levels (i.e., how likely students were to succeed on the levels) demonstrates a similar conclusion, showing a significant, medium sized increase in performance using a paired-samples t-test, $t(173) = -4.96$, $p < 0.001$, $d = 0.51$.

While these initial pre-posttest results are encouraging, the replay data can be leveraged to see if there was finer grained behavioral evidence that players not only got better at the game but did so because they were better instantiating the principles it targets. This result would mean that before and after playing the game for some time, players would build towers in the unguided pre-posttest levels (i.e., levels without the constraints of energy dots, allowing more freedom in construction) that showed a better awareness that (1) a structure with a wide base is more stable, (2) a structure with a lower center of mass is more stable, and (3) a structure that is symmetrical is more stable. It is important to note that looking at a difference in metrics related to learning goals is different from looking at the difference in player success rate. If we entertain the possibility that the game is not necessarily well aligned, then it is possible that players could improve in their pass rate in the game for reasons other than following the principles that are central to the game's goals.

To test whether players were better leveraging the physics principles targeted by the game in their solutions, I built an *Interpreter* for the RAE to calculate a variety of metrics based on each player's final state of each in-game pre-posttest level. These metrics represent the PRMs in *RumbleBlocks* and are defined as: the width of the tower's base (Figure 4A); the height of the tower's center of mass, calculated as a weighted average of the centers of mass of the tower's blocks relative to the ground (Figure 4B); and a measure of symmetry defined as the size of the angle formed by a ray from the center of the tower's base to the center of mass and 90° (Figure 4C). Once I had a set of metrics, I normalized the scores across players within each level to have a mean of 0 and a standard deviation of 1. Normalization was done to account for differences in scale between the levels (e.g., some levels have a higher target zone for the ship resulting in higher centers of mass for all towers on that level) which made it difficult to compare metrics directly between levels.

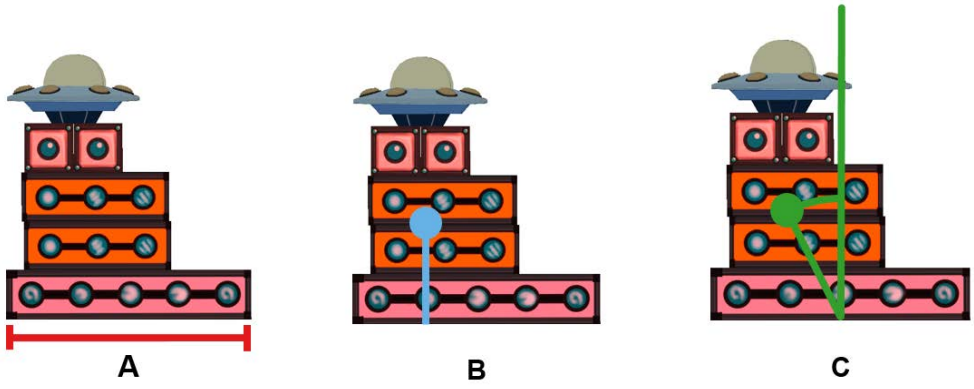


Figure 4. A visual depiction of each of the 3 Principle-Relevant Metrics used in the analysis of *RumbleBlocks*. (A) Base Width, (B) Center of Mass Height, and (C) Symmetry Angle.

To see if there was any improvement on the use of principles in players' solutions from the pre- and posttest levels, I compared each student's averaged PRMs using repeated-measure ANOVAs. Looking at the results in Error! Reference source not found., I saw a significant improvement for the Base Width, $F(1, 253) = 9.31$, $p = 0.003$, and Symmetry Angle $F(1, 253) = 5.94$, $p = 0.016$ metrics, meaning that at the end of playing the game,

students were beginning to design towers that had wider bases and more symmetrical layouts. However, I did not see any significant difference in terms of Center of Mass (COM) Height, $F(1, 253) = 0.35$, $p = 0.552$, meaning that students did not seem to attempt to lower the center of mass of their structures. This result suggests that the original version of the game may possess a misalignment in how it handles the low center of mass principle; however, as stated previously, a null result in pre-posttest comparison cannot reach this conclusion definitively.

Metric	MS	*df*	*F*	*p*	
Base Width	3.37	1	9.31	0.003	**
COM Height	0.14	1	0.35	0.552	
Symmetry Angle	1.94	1	5.94	0.016	*

Table 1. Repeated-measures ANOVA results for average normalized Principle-Relevant Metrics from the in-game pre- and posttest in the formative evaluation study of *RumbleBlocks*.

Alignment Regression Analysis

Knowing from the pre-posttest analysis that there were possibly some misalignment issues with *RumbleBlocks*, the next step in analysis was to explore why this might be happening. This result led me to explore the question: is the game properly incentivizing players to act in a way that corresponds to the pedagogical goals for the game? If the game is knocking over towers that are principled or letting unprincipled towers remain standing, then players will not know what to make of the feedback they are given, making it unlikely for them to improve toward better understanding. Such cases would be examples of misalignment.

To answer this question, I turned to my definition of alignment as a regression between PRMs and game feedback. In effect, I needed to test if there was a relationship between the relative principled-ness of student solutions and whether the game deemed the solutions successful. Put another way, I needed to establish whether the principled-ness of a tower (measured by the relevant PRM) could predict that a tower would stand or fall in the earthquake. To facilitate this analysis, I augmented the original *Interpreter* to have the RAE calculate the same PRMs I used for the pre-post analysis, except this time for all levels. I wanted to explore how well the metrics that should indicate a well-constructed tower (i.e., a domain-principle-based separation function of the solution space) corresponded to a player passing a given level (i.e., a feedback-based separation function of the solution space). It is important to note that this analysis is concerned primarily with the behavior of the game and not with student performance. In this context students are merely providing the test data for my analysis of the game's system.

In measuring alignment throughout this document, I employ a hierarchical mixed-effects regression method[14]. We would expect from this analysis that, as a player creates towers that exhibit stronger adherence to each of the

[14] Note: In previously published versions of this work, slightly different forms of regression and coefficient scaling have been used. In [62] regression was performed with values scaled in terms of the maximum or minimum values for a given level and done within groups of levels targeting the same principle. In [60] a similar approach was taken with regard to grouping levels but with coefficients normalized (i.e., mean=0, SD=1) instead of max scaled. Given that many more regression models are being run and compared in the current work and previous analyses suggested we should be skeptical of the principle target labels for some *RumbleBlocks* levels [59], I have opted for the simpler paradigm used here. My conclusions have always been generally similar regardless of the specific approach to regression.

target principles of the game, they would be more likely to succeed on the game's levels. The regression model I use to evaluate alignment takes the following form:

$$Outcome \sim COM\ Height + Symmetry\ Angle + Base\ Width + (1 \mid Level)$$

Equation 1. Hierarchical regression model used for evaluating alignment.

In this model, $Outcome$ is a binary outcome of whether the tower succeeded on the level; $COM\ Height$, $Symmetry\ Angle$, and $Base\ Width$ are the PRMs as defined in Figure 4; and $(1 \mid Level)$ denotes a random intercept for Level, meaning that some variance in outcome is expected due to the idiosyncrasies of each individual level design. Coefficients from the model are reported in both raw form (B) with standard errors (SE B) and in standardized (β) form. The raw coefficients represent the effect of a single unit increase of the given metric's natural scale, while the standardized coefficients represent the effect of a single standard deviation increase from the given metric's mean. In general, the raw coefficients are better for understanding the strength of an individual metric's contribution where the standardized ones are better for judging the relative strengths of the metrics compared to each other.

To aid interpretation, raw COM Height and Base Width scores are scaled such that a one-unit increase in the measure is equivalent to the size of one square block (the grey squares visible in Figure 3), which is the smallest block size in the game. Further, in all regressions throughout this document, COM Height and Symmetry Angle are both reverse-coded so that a larger coefficient value represents a better alignment between game goals and game outcomes.

Given that Outcome is a binary variable, the model is fit as a logistic regression. As a reminder, when interpreting logistic regression models, the coefficient B_i or β_i correspond to the change in the log of the odds for a one-unit change in factor x_i given that all other factors remain fixed. The change of odds ratio (OR), i.e., the change in percentage chance that a tower succeeded, for a 1-unit change of factor x_1 can be computed by raising e to the power of the coefficient, $OR = e^{\beta_1}$.

The results of applying this regression process to the formative evaluation data can be found in Table 2. When looking at the PRMs for Base Width and Symmetry Angle, there is a significant relationship between the PRM and success on the level, which is what would be expected if the game is appropriately incentivizing their target principle. The relationship for the COM Height PRM, however, was not found to be significant. This would mean that, counter to what the target principles suggest, players who build with lower centers of mass are not any more likely to succeed on levels than players who build towers with higher centers of mass. This behavior could not have been the *RumbleBlocks* designers' intent.

Coefficient	B	SE B	β	p	
(Intercept)	1.077	0.450	1.749	-	
Base Width	0.136	0.031	0.188	< 0.001	***
COM Height	-0.049	0.047	-0.039	0.295	
Symmetry Angle	0.050	0.005	0.321	< 0.001	***

Table 2. Alignment regression results for the formative evaluation of *RumbleBlocks*.

Discussion

The results of the initial formative evaluation of *RumbleBlocks* suggest that the game did have learning gains, particularly for the symmetry and wide base principles, but had issues with the center of mass principle. In looking at the relationship between students' solution features and their likelihood of success on a given level, I found that

the pattern in student learning appears to be replicated in the coefficients of the metrics predicting success, in that more principled Symmetry Angles and Base Widths are associated with a higher likelihood of success on a level, while a more principled COM Height was not associated with success (and was in fact trending negative).

I interpret these results to suggest that *RumbleBlocks* has a problem with its alignment. As players are going through the game it would be hard for them to associate having a lower center of mass as a solid structural principle because the game does not provide consistent feedback to make the connection. Given that the pre-posttest results show a similar pattern as the alignment regression, it is clear this misalignment is an issue with the game's design that needs to be remedied with design refinement.

One might be tempted to suggest that if there is such a misalignment with the center of mass principle, then the designers should simply alter the game to base feedback directly on that metric. This suggestion is not an unreasonable idea; but in the case of *RumbleBlocks,* it would be difficult to implement. For one, while the design of each level in *RumbleBlocks* is intended to emphasize one principle over another, the principles can never be truly separated from each other. Making it so that success for a level targeting one principle is determined by the respective PRM might cause that level to become misaligned for another principle. Further, these principles are high-level abstractions for the dynamics of real earthquake physics, which possesses some inherent stochasticity that cannot be estimated in a closed form (see Appendix C for further details on this issue).

More broadly, I find this work provides encouraging evidence for my formulation of alignment. Given that there was a misalignment of the COM Height metric and success in the game, it stands to reason that players would not be building with lower centers of mass from pretest to posttest, and this pattern was, in fact, observed. If alignment regression corresponds to learning results, that suggests that it could be used as a valid assessment of a game's fitness to its instructional goal.

Further, this work was a first demonstration of Replay Analysis. In this instance, I employed the system both for measuring learning in terms of pre-post gains but also to evaluate the alignment of a game and its goals. Both of these helped to broaden the understanding of how *RumbleBlocks* worked. As the case study continues to unfold in later chapters, the capacity for Replay Analysis to answer even more questions from this same dataset will be demonstrated.

CHAPTER 6 EXPLORING SOLUTION SPACES IN RUMBLEBLOCKS

In the previous chapter, I provided an initial demonstration of Replay Analysis in service of measuring alignment. From the perspective of Schön's seeing-moving-seeing framing of the design process [138], the evaluation so far amounts to a first glance at a particular context. The fact that Replay Analysis can provide this kind of first pass appraisal of a game's fitness is good, but there is potential for it to be used further. In the event that an issue is found at an abstract level, the next question for a designer is to try to better understand why it might be happening. Exploring this question requires reframing the design context [48] and viewing it from different angles to suggest what solutions might lead to a better design. Replay Analysis also affords this kind of exploration, as I will demonstrate in this chapter.

The fact that *RumbleBlocks* is not providing feedback in accordance with its pedagogical goals gives rise to a new question: if players are not getting consistent feedback on the center of mass principle, what is the game doing in these situations? Answering this question requires the ability to look at players' experiences in closer detail than is provided by distilled metrics. Leveraging the Replay Analysis approach provides the ability to look at the common structure of solutions that the players created and judge whether the game's reactions to those solutions was appropriate.

Attacking this question required wrestling with the issue that neither the designers of *RumbleBlocks* nor I had a sense of how many ways there were to solve any of the levels in the game. Further, among the various ways to solve each level, which ones are the young players of the game's target demographic more likely to employ? Resolving these issues would enable looking at the kinds of feedback the game provides to common solutions and whether that feedback appears to make sense given the principles of the game. Rather than comb through myriad solutions to each level myself, I made use of the RAE to reframe the log data as a detailed representation of the solution space. This representation could then be coerced into a form that would make it amenable to various machine learning methods to cluster structurally similar solutions.

At a high level, this analysis involves capturing a picture of the space of solutions that students use to overcome in-game challenges. This solution space is generated by clustering individual solutions created by students into a subset of representative solutions (since there are too many to view individually). Once a collection of representative solutions is gathered, each one is evaluated in terms of its PRMs. Finally, the PRMs are compared to the positive or negative feedback designation that the game's mechanics assigned to the majority of individual solutions embodied by each of the representative solutions. This process allows me to analyze the general principled-ness of a set of student solutions and how the game treated them as a means of evaluating alignment.

An alternative way to approach this problem could have been to employ some kind of AI to exhaustively explore all possible solutions to in-game levels and challenges, and there is some precedent for approaching design problems in this manner [147]. I focus instead on the analysis of representative solutions for three main reasons. First, the generation of an exhaustive space could be infeasible for games with lots of variation in solution approach, particularly during prototyping stages where minor changes might have drastic effects on possible solution space. Second, many existing techniques for exhaustive game space exploration require formal description languages such as the Videogame Description Language [134]. Relying on the conclusions of an abstract representation of a game may elide over idiosyncrasies or hidden assumptions that would arise in a realized prototype put before players. Finally, focusing on patterns across the behavior of actual players cuts right to the heart of the reason for evaluating a design in the first place. Understand how players will approach and explore the game is why playtesting is conducted and focusing on their behavior allows one to focus on this mission.

Conceptual Feature Extraction

The complex structure of towers in *RumbleBlocks*, and games more generally, makes their representation inappropriate for common clustering methods. To perform clustering, I first had to develop a means of describing solutions in terms of their structural features in a way that machine learning methods could understand. For example, many students might build a tower that uses an arch pattern, whereas others might build an inverted "T" shape. I needed a representation that captured elements of these basic structural patterns. This effort lead colleagues and I to the development of the Conceptual Feature Extraction (CFE) algorithm, previously presented in [61], which leverages a two-dimensional grammar induction process to characterize solution states within *RumbleBlocks*.

At a high-level, the Conceptual Feature Extraction process flows through a series of four stages (illustrated in Figure 5). First, the raw game representation of the solution state is discretized to a grid making use of the objects' physics collider properties. Next, an exhaustive two-dimensional context free grammar is learned over all of the solutions in a given dataset. This grammar can then be used to parse the set of solution states, and the resulting parse trees can be flattened into binary feature vectors that are amenable to clustering.

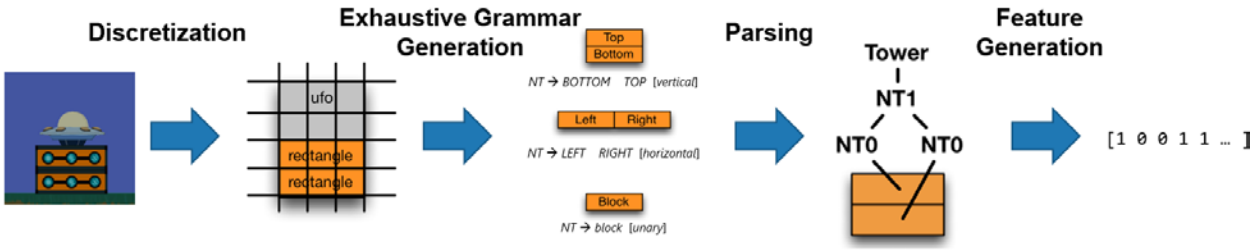


Figure 5. A high level description of the Conceptual Feature Extraction Process.

To build this new representation, I developed a new *Interpreter* for the RAE to produce representations of student towers aligned to a two-dimensional grid. The RAE made use of a common physics formalism available in the Unity game engine called an axis-aligned bounding box, which describes the physical extents of an object aligned to each of the three principle axes at any given point in time. Relying on this formalism means that while this approach was largely developed with *RumbleBlocks* in mind, it would be applicable to any game relying on a similar physical structure.

Next, two-dimensional grammar induction learns a set of patterns that can be used to describe all the student solutions in the entire dataset. A two-dimensional grammar consists of three components:

1. Terminal Symbols, which represent the blocks, spaceship, and empty space (in this context)
2. Non-terminal Symbols, which represent structural patterns consisting of more than a single block
3. Rules, which map non-terminal symbols to pairs of other non-terminal symbols oriented in a certain direction (horizontal or vertical), or non-terminals to terminal symbols (a unary relationship).

To help illustrate the concept, Figure 6 shows a simple example grammar (u, h, and v represent unary, horizontal, and vertical respectively), and the parses for two simple towers.

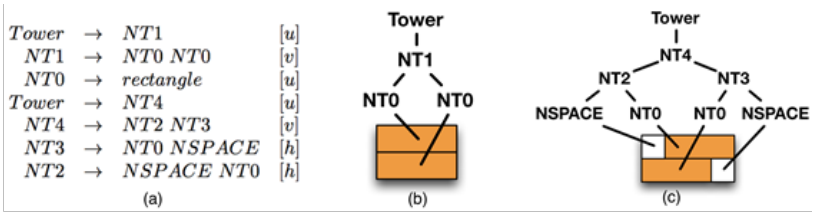


Figure 6. A simple two-dimensional grammar (a) and the parse trees generated by applying this grammar to two towers (b and c).

CFE first generates an exhaustive set of rules that describe every possible way to parse all of the solutions in the set. Next, it computes all the possible parses of each solution. Given the parses for each solution, it creates a hot-1 encoded vector (i.e., containing a 1 for every non-terminal present in the solution and a 0 for every non-terminal not present in the solution) for each solution. The resulting feature vectors contain information about all the structural patterns present in each solution. These patterns may correspond to individual blocks, pairs of blocks, more complex combinations of blocks, or even whole towers. The vectorized patterns can then be used for a host of machine learning applications, including clustering and classification.

Using CFE to Consider Solution Spaces

Given a set of solutions described in terms of their common structural features with CFE, the next task is to create a set of representative solutions that can enable designers to look at the game's behavior across multiple player experiences. I frame this task as clustering, where each unique cluster in a solution can be taken as a representative solution to a level. Using clusters, an archetypal solution can be considered as the aggregate of the solutions contained within a given cluster.

In creating representative solutions for *RumbleBlocks,* I performed clustering across the levels of the game. For each level, I clustered the vectorized solutions using g-means, a variant of the common k-means clustering algorithm that chooses a value for k optimizing for a Gaussian distribution within clusters [57]. This process produced a set of different groups for each level, where each group represents solutions that share structural similarity. The resultant clusters can be summarized as representative solutions that embody the general trend within the cluster. For each cluster, I created a representative solution by averaging the PRM scores within the cluster and then assigning the success label that the game assigned to the majority of solutions within the cluster. Aggregating solutions in this way gives me the ability to think about common patterns of solutions through a single representative solution rather than individual solutions.

To get a sense of the general trends in how the game treats different solutions to a particular level, I created representative plots of the clusters like the one show in Figure 7. These plots show each representative solution plotted with its frequency of use (as a percentage of all observed solutions for that level) along the x-axis and its relative principled-ness, in terms of a normalized PRM score, along the y-axis. The green squares represent solutions that are mostly successful where the red diamonds show solutions that are mostly unsuccessful. When examining these plots, two different patterns are primarily of interest: principled failures and unprincipled successes, which both represent the game generally giving feedback contrary to the target principle for the particular level. These cases can shed light on potential problems with a game's alignment. The analysis of *RumbleBlocks* highlighted several cases, but for purposes of this document, I only discuss two in detail.

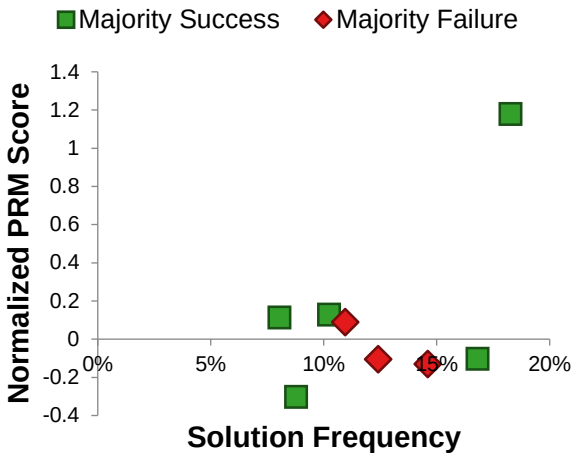


Figure 7. A plot of representative solutions' PRM score versus frequency.

The first problem level is Symmetry_7, which is a level meant to target the concept that a symmetrical structure is more stable. In this example, there are two highly frequent solutions, the two points farther to the right in Figure 8. One is mostly successful and the other is mostly unsuccessful, but they do not differ strongly in their PRM scores for symmetry. When I examined screenshots of student solutions to this level, I saw the situation shown in Figure 8, where the tower on the left (an inverted T-shape) comes from the majority failure solution while the tower on the right (an arch shape) comes from the majority success solution. While it is clear from the examples that the left tower should fail (as it did frequently), it is important to remember that this level is designed to target the symmetry principle, which says a symmetrical structure should be more stable. Both solutions seen in these representative solutions are generally symmetrical, but one was considered a failure while the other is considered a success. This case represents *RumbleBlocks* giving inconsistent feedback to players about the targeted symmetry principle. An alternative interpretation is that this level should not be labeled as targeting the symmetry principle, given that its two most frequently used solutions both embody a reasonable level of symmetry.

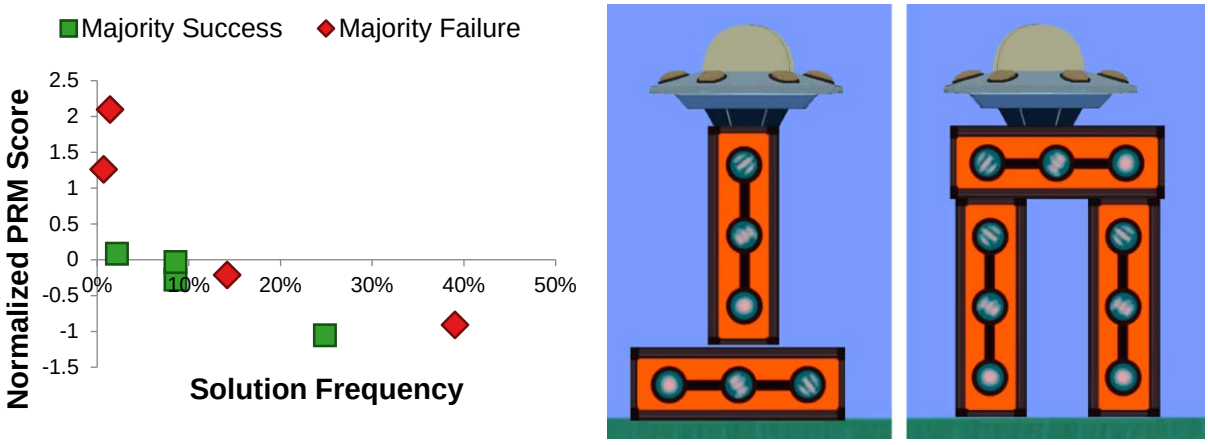


Figure 8. A plot of solution frequency (as a percentage) vs. PRM score for all of the clusters on the Symmetry_07 level of *RumbleBlocks* (Left) and two example student solutions to the Symmetry_07 level. The solution on the left comes from a majority unsuccessful cluster (Right).

Another anomalous example is shown in Figure 9, which shows a plot of the different solutions to the level CenterOfMass_10_PP. This level was used as part of an in-game pre-post design, meaning it omits the energy dot mechanic and is based on a level designed to target the low center of mass principle. It is harder to attribute patterns in the chart to elements of level design because it lacks the energy dot mechanic and thus does not restrict players as much as normal game levels; however, an interesting pattern develops nonetheless. The distribution of how many students created each solution on this level is more evenly spread; but among groups of solutions that are all relatively equal in PRM score, we see two solutions that are majority failure rather than success.

Visually inspecting the solutions students created to this level, we see the pattern that arises in Figure 9, where an example from one of the successful solutions is shown on the top and an example from each of the unsuccessful solutions is shown on the bottom. The salient feature to note among the unsuccessful solutions is the presence of the alien's spaceship on top of a single square block. This pattern points to a nuance in the game's mechanics, where the criteria for in-game success is whether the spaceship falls off the tower during the earthquake and not just that the tower continues to stand. Such a pattern opens up the possibility, illustrated by the lower right quadrant of the matrix in Figure 1, that a student could build a perfectly reasonable tower that is judged as unsuccessful by the game only because the spaceship falls off. This behavior is an example of the more nuanced kind of alignment issue where a task requires an extra piece of unexpected knowledge to complete the level successfully. While the spaceship should also be subject to the same stability principles as any other part of the tower, this level seemed to suggest it was the major determining factor in success, disregarding the rest of the tower's design.

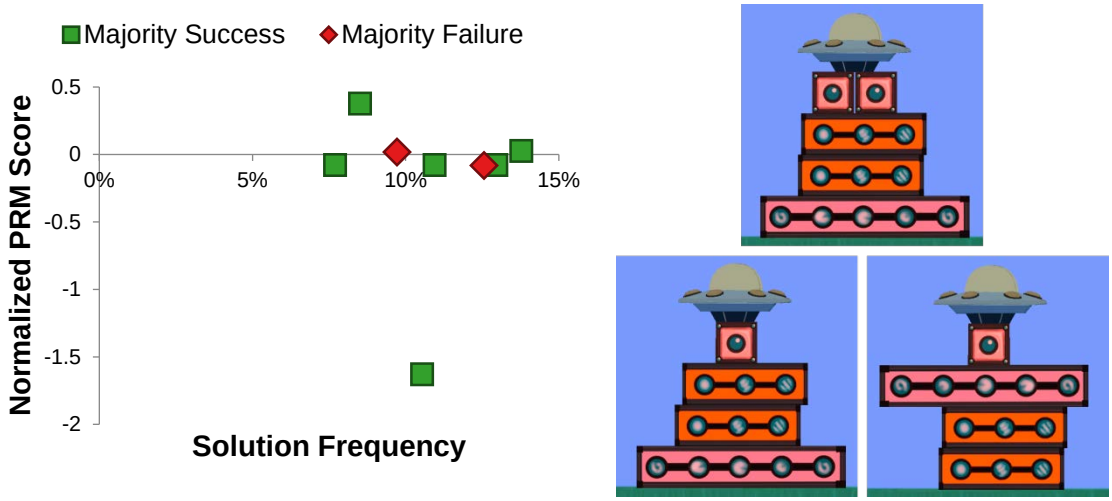


Figure 9. A plot of solution frequency (as a percentage) vs. PRM score for all of the clusters on the CenterOfMass_10_PP level of *RumbleBlocks* and examples of student solutions on the CenterOfMass_10_PP level. The solution at the top comes from one of the majority successful clusters.

The patterns I observed in the analysis of the Symmetry_7 and CenterOfMass_10_PP data were present in a number of other levels as well. As a pattern of salient features emerged, I wanted to see if there was further evidence in the structural data to support the conclusion that *RumbleBlocks* might have an issue with certain structural features exerting too much influence on the success of a tower. To explore this question, I used the structural features generated through the CFE process and used a χ^2 analysis to identify which structural features present in student solutions were more predictive of success. I performed a χ^2 test of each of the 6,010 symbols

against solution success to see which patterns were most strongly related to success of a tower. Because there are a large number of statistical tests involved, the possibility of a false positive (i.e., Type I error) is increased. I applied a Bonferroni correction to the results to account for the number of statistical tests [49]. This correction divides the cutoff for considering a result to be significant (conventionally $p < .05$) by the number of tests performed (in this case 6,010) and uses the result (8.32e-6) as the new bar for significance.

Overall, 19 grammar symbols were significantly related to success. However, representing the grammar symbols with grounded game objects resulted in only five distinct structures[15]. Once I had a selection of significant features, I performed a logistic regression of those features against solution success to understand the direction of the relationship (i.e., does each feature predict success or failure) as a χ^2 can only detect the presence of an association and not its direction. I only present the directional results of this regression and not the actual coefficients. The results of this process are visually rendered in Figure 10.

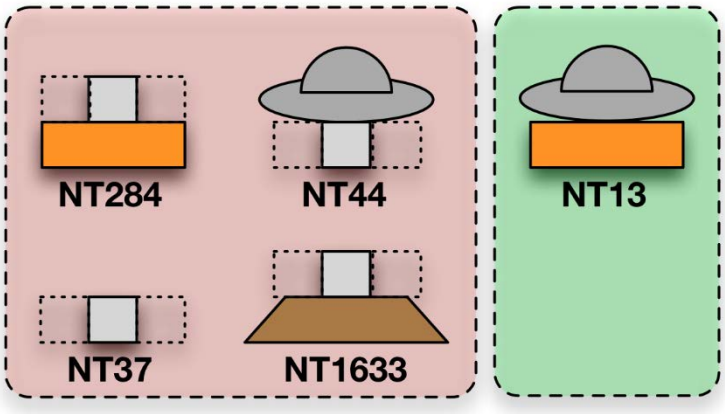


Figure 10. Rendered results of a χ^2 analysis of structural features in *RumbleBlocks* which predict the success of a tower in the earthquake. Student solutions that contained the features in the failure region to the left were more likely to be unsuccessful in the earthquake, while solutions that contained the feature in the success region to the right were more likely to be successful.

The original question driving this exploration asked: if players are not getting consistent feedback in the game, what is the game doing? The pattern that arises from the χ^2 analysis demonstrates that the game tends to focus more on points of weakness with a lone square block without supports, meaning that the game is generally punishing more nuanced sub-structural faults of towers. The principles targeted by *RumbleBlocks* are generally meant to apply to whole structures, and so do not necessarily account for these kinds of smaller structural problems. These results would suggest some misalignment between the stated goals of the game and its feedback mechanisms as instantiated in the earthquake mechanic.

Further, the analysis demonstrates a strong difference between the width of the platform underneath the spaceship and the eventual success of the tower (i.e., placing the spaceship on a single block is more likely to lead to failure and placing it on a wide block is more likely to lead to success). This pattern highlights the importance of the spaceship remaining on top of the tower as a key success criterion. While the designers were aware that

[15] The CFE process uses a set of recursive rules to represent negative space, allowing it to create multiple symbols that would look visually the same but differ in how negative space around the tower is handled. More details on this limitation can be found in the paper on CFE [61].

the spaceship served such a purpose in the design, they probably did not think it would be such a strong determining factor to the potential detriment of other learning goals. When pursuing iteration, the designers of *RumbleBlocks* will have to consider if this result represents a flaw in the game's mechanics, which contradicts the message, or an opportunity to teach a nuanced aspect of stability and balance with some new feedback.

Limitations of CFE

CFE has enabled a number of analyses that have highlighted potential issues with *RumbleBlocks.* While the process has been useful in this case there are several limitations to the CFE approach that would make it difficult to generalize to other contexts and games and are worth noting.

The main limitation of CFE is its reliance on a two-dimensional context free grammar formalism. While the axis-aligned bounding boxes I use in the process are common in many game engines and it is relatively trivial to expand the approach to a three-dimensional space, it is still restricted to primarily spatial relationships between objects that can be fit to a grid. Games whose spaces are less obviously geometric would struggle to adapt the approach. Further, the current grammatical approach cannot describe all geometric patterns itself. For example, a spiral pattern like the one in Figure 11 cannot be describe using a two-dimensional context free grammar because there is no way to cleanly separate any of its elements along a single axis.

Figure 11. An example of a spiral pattern that a two-dimensional context free grammar could not encode.

Viewing CFE more generally as a grammar induction strategy for encoding game states it is possible to envision other grammatical formalisms besides two-dimensional adjacency that could be used for non-geometric games (e.g., applying natural language processing techniques to a game whose space it primarily textual responses). This perspective, however, places an additional burden of feature engineering on game designers to develop new grammatical rules for understanding their games.

Further, CFE is a brittle process meaning that as new playtesting is conducted there is no guarantee that an existing grammar could describe the new solutions. In order to perform analysis in an ongoing fashion a new grammar has to be re-learned on the entire body of player solutions each time. This requirement can make it difficult to translate findings between iterations of design as each iteration is being interpreted under its own grammar rather than a common one between all iterations.

Addressing these limitations of CFE was part of what motivated colleagues and I to develop the TRESTLE concept formation algorithm [93]. Rather than take the approach of encoding game states as feature vectors to apply mainstream machine learning techniques, TRESTLE was developed to take examples described in the common JavaScript Object Notation (JSON) and organize them into a conceptual hierarchy. The algorithm is also inspired by cognitive research on human category learning [50] and so develops concepts incrementally allowing it to incorporate new data as it is collected. Leveraging these features could allow TRESTLE to serve as a more general way of clustering player solutions to explore solution space. For more specific details on the implementation of TRESTLE see Appendix B.

To demonstrate TRESTLE's capacity as a clustering technique for exploring solution space I compared clusterings produced by TRESTLE and by CFE to groupings produced by humans to see how well they agreed. For this analysis, I created an *Interpreter* for the RAE that produced screenshots of player solutions to three of the pre-posttest levels of *RumbleBlocks*, as well as state descriptions that could be parsed by TRESTLE and CFE. The pre-posttest levels were used because they are likely to have some of the most interesting variance in solutions as they were minimally constrained. Once the data was generated I had two assistants produce groupings of the screenshots using an open-card sorting process (i.e., there was no previously agreed upon number of groups). I then applied both CFE and TRESTLE to cluster the solutions. The clustering process for CFE worked as I described above, while TRESTLE applied to process described in Appendix B optimizing for the Bayesian Information Criterion (BIC) [139] heuristic.

The clusters created by each method were compared to the human grouping using Adjusted Rand Index (ARI) [124]. ARI is a measurement of agreement between raters where the raters do not need to agree on a number of groups and can be viewed as a generalization of Cohen's Kappa [163]. The scale of ARI ranges from -1.0 to 1.0 where 1.0 represents perfect agreement, -1.0 represents perfect disagreement, and 0.0 representing chance. Because the g-means and TRESTLE algorithms have some stochastic processes, clusters were generated 10 times for each level averaging ARI across the runs.

Level	Clustering	ARI	STD
Level 1	**CFE**	0.51	0.08
(n=251)	**TRESTLE**	0.51	0.02
Level 2	**CFE**	0.47	0.04
(n=249)	**TRESTLE**	0.53	0.05
Level 3	**CFE**	0.42	0.02
(n=254)	**TRESTLE**	0.49	0.06

Table 3. Average and Standard Deviation (STD) Adjusted Rand Index (ARI) measures for CFE and TRESTLE across three levels of *RumbleBlocks*.

The average ARI measures of cluster agreement can be seen in Table 3. The results of the comparisons show that TRESTLE demonstrates equal or better agreement with human groupings than CFE does across the three levels. TRESTLE's clusters are at least as good as ones created by CFE but they have the benefit of using a less constrained representation for game states that would enable application to a wider set of games. Further, TRESTLE, being an incremental algorithm, is less brittle in handling new data from new playtesting allowing for analysis to continue within a common understanding.

While this analysis demonstrates that TRESTLE could also serve as a potentially better clustering approach for solution space analysis, the algorithm does lack the ability to support the kind of sub-structural pattern analysis demonstrated by the χ^2 results. This characteristic highlights the importance of being able to look at player experience from many different angles and perspectives to understand how a game is being played. A Replay approach readily affords this ability.

In the context of the broader thesis, this chapter highlighted the capacity for Replay Analysis to enable reframing of a game to understand it from different angles. As part of this work, I developed several techniques to enable clustering of solutions in game levels. Considering solutions in terms of clusters enabled me to define the solution space of the game in terms of representative solutions existing in the original log data. Being able to see this solution space and frame it in several ways allowed me to move from a general sense of misalignment to a concrete behavior pattern that could be addressed with a specific design change.

CHAPTER 7 CLOSING THE LOOP WITH PROJECTIVE REPLAY

The work I have described so far has demonstrated the power of Retrospective Replay Analysis to explore the alignment of an educational game using observed player behavior. The approach enabled both alignment evaluation and an exploration of the solution space from multiple perspectives using a single common dataset. In the context of *RumbleBlocks,* these analyses highlighted a general problem of alignment and concrete potential issues that could be addressed in iterating the game's design. The next logical step in the design process would be to explore whether fixing the highlighted problems results in a demonstrably better game. In the educational data mining and learning analytics literature, this type of study is called a close-the-loop evaluation [35,79], and these evaluations are exceedingly rare. The goal of the work in this chapter is to demonstrate how an extension of Replay Analysis, namely Projective Replay Analysis, can facilitate close-the-loop evaluations that would provide designers with similar insight to formal student playtests and further to perform such an evaluation with the results of the *RumbleBlocks* alignment analyses.

Concretely, the primary question addressed in this chapter is: would a Projective Replay Analysis of a redesigned game come to the same conclusions as a full close-the-loop evaluation? To explore this question, I performed Projective Replay Analyses of several design variations of *RumbleBlocks* and a new close-the-loop study of one of the variations. I then compared how well the results of the Projective Replays predicted the results of new human playtesting in order to evaluate whether Projective Replay could be a suitable evaluation of a game design on the same level as additional human playtesting.

Additionally, this chapter addresses a second question: do redesign ideas based on alignment-focused analytics produce a better aligned game? To answer this question, I compared how the alignment of PRMs to in-game feedback and the shape of the solution spaces generated by the various Projective Replays relate to the original game design. The goal with this analysis is to demonstrate that following the conclusions of alignment-focused analytics produces an upward trend in alignment.

This chapter will be organized as follows. First, I describe the origins of the three design variations that were considered for testing and detail the rationales behind their design. Second, I detail the method I developed for characterizing the relationship between two solution spaces in order to understand how a particular design choice altered the space of what players did in the game. Next, I present the results of Projective Replay Analysis on each of the candidate redesigns followed by the results of the close-the-loop classroom study and a discussion of how its conclusions relate to those of the Projective Analyses. Finally, I discuss the implications of the results on Projective Replay Analysis as a valid evaluation method of game design variants.

Design Variations of RumbleBlocks

In order to understand whether Projective Replay Analysis can approximate the results of human playtesting with new versions of a game, I required multiple variants on which to run Projective Replay Analysis. In consultation with practicing game designers and based upon the alignment and solution space evidence so far, I developed three mechanical variations for *RumbleBlocks*: Glue Blocks, No Cliff, and No Ship. In this section, I describe the rationale and implementation behind each of these variations.

The Glue Blocks Variation

One of the bigger lessons from the solution space analysis is the results of the χ^2 analysis shown in Figure 10, where the success of a tower is more strongly related to micro faults within the tower than with broader aspects of the tower's design. A way to address this problem of micro faults is to explore a version of the game where blocks act together as a single rigid body, so that feedback is likely to be better aligned with the broad principles

that the game is meant to target. A design that involved connected structures was considered in the preliminary design phases of *RumbleBlocks*[16], but initial prototype testing indicated that players found the disconnected structures to be more fun in the game. The designers also thought that having a disconnected block mechanic would allow for more interesting dynamics in the design. However, if we introduce a mechanic that allows players to glue blocks together so that the blocks act more like connected structures, the level may be better modeled by the principle-relevant metrics.

To implement this variation, I added a mechanic that connects the blocks of a tower before the earthquake begins. I do this by calculating the tower's aggregate center of mass as a weighted average of the centers of mass of each of its blocks. I then instantiate a new GameObject at that position with the aggregate mass of the tower. The blocks of the tower are attached to the new GameObject as children and have their Rigidbody components deactivated so they no longer simulate their own physics. Effectively, this change creates one large object that has the collider profile (i.e., boundary silhouette) of the stack of blocks and the mass of all of the blocks added together, acting like a single rigid body in the shape of the tower. I did explore implementing the mechanic by attaching the blocks of a tower to each other using Unity's physics joint system; however, this approach caused problems in the physics engine's constraint solver because there were too many fixed joints to solve, which caused errors when the system could not process them all.

The No Ship Variation

Another interpretation of the χ^2 analysis suggested that which block(s) supported the ship had an outsized effect on the outcome of a level. This pattern highlighted the spaceship having an outsized effect on the success criteria such that the real goal of the game is to stabilize the ship and not to build a stable structure. Another possibility for changing the mechanics could be to remove the spaceship as a factor entirely. If success was not defined purely by the spaceship, players would have to think about a broader sense of stability in their towers. Removing the spaceship success mechanic could also preserve the interesting dynamics of having disconnected structures, which were valued by the designers.

There are some design challenges to removing the spaceship, however, as it serves several purposes in the game. Currently, the spaceship is how players submit a solution (i.e., placing the ship on top of the tower). The spaceship also serves a narrative purpose as returning the ship to the alien is currently the motivation of the game. Further, the ship is also cleverly used as an early tutorial for how to rotate objects because the ship always starts upside down, requiring rotation before submitting the first level.

Rather than remove the spaceship from the game entirely, I designed a change to the scoring mechanics in an attempt to limit its outsized impact on success. Currently, success is defined by the ship remaining in its target zone after the earthquake. In the No Ship variation, the success criterion is changed to simply holding on to the energy dots through the earthquake. When the earthquake is started, the ship's collider is turned off (so it is ignored by the physics system) and it goes into an animation to slowly float upward. As the earthquake shakes, the system keeps track of the number of active energy dots captured by the blocks during every physics update frame. Once the quake is finished, if the average number of dots across all the frames of the earthquake (rounded up) is equal to the number of dots in the scene, the player succeeds and moves on the next level; otherwise, they fail. The goal behind this design is to require players to pay attention to the overall stability of their structures rather than focusing on the spaceship.

[16] http://www.etc.cmu.edu/projects/illuminate/?page_id=221

The No Cliff Variation

In developing design variations for this study, I consulted with several practicing game designers, showing them the alignment and solution space results so far and asking what they might try to fix. Several interesting candidates resulted from this process. For example, several ideas related to the concept of the alien being worried about their ship, appearing more nervous if it shook too much or more confident if it stayed more stable throughout the quake, as a way of providing a more continuous feedback signal. Another idea was to change the shape or size of the spaceship as another parameter to adjust for challenge.

One of the more promising ideas was the simple suggestion of removing the cliff from the equation. The designers noticed some examples of solutions that seemed to use the cliff as a crutch, which possibly contributed to allowing unprincipled towers to stand. They reasoned that there is no real functional purpose for the cliff to have a physical presence in the scene, and it could simply be a background visual element. A similar issue of the cliff holding up towers did emerge in earlier design iterations of *RumbleBlocks*, resulting in the shape of the cliff being altered but leaving its physical presence intact. To implement this No Cliff variation, the game simply deactivates the collider of the cliff once a level starts so that it will not have a physical presence in the scene. It is notable that, unlike the other variations, this one is activated at the beginning of the scene, meaning it could affect how players build their towers and not just how they are treated by the earthquake.

Solution Space Shift

While the paradigm of Projective Replay Analysis has the potential to guide closed-loop design iteration, there is one fundamental concern with the approach; namely, if the game is behaving differently, then the possibility exists that players will play differently. This concern would seem to imply that the instant a game's mechanics are changed, any replay recordings taken from the old version of the game become invalid. I do not dispute this notion in general. I would, however, argue that changes in player behavior due to changes in game behavior are a matter of degree rather than absolute. Depending on the nature of the mechanical change, players might be able to have new experiences than they could have previously, or a change could effectively outlaw previous player behavior by making it impossible. To better understand this relationship between a game design change and a change in players' experiences, I developed a way to characterize the relationship between solution spaces. By looking at how a game's original solution space relates to the one afforded by a new variation, designers can revise their intuitions about how well the game suits their goals.

Concretely, I suggest that any change to a game's mechanical structure will have one of five possible effects on its solution space:

1. The new solution space could remain the same. This outcome would mean that the affordances available to players to explore the solution space are no different than they were before. While the structure of the space remains the same, the change could cause existing solutions to receive different feedback, leading the game or the players to interpret a space differently. Even if the structure of the space does not change, it is still possible that alignment could change.
2. The new solution space could be a subset of what it was before. In this case, previously reachable solutions are no longer possible. This outcome would be desirable in cases where specific solutions or mechanical uses were found to foster misconceptions.
3. The new solution space could be a superset of what it was before. This outcome would mean that the game now affords more variation to players than it originally did. In such cases, it is necessary to evaluate the alignment of this new territory of the solution space to ensure that educational goals are still being met.

4. The new solution space could be a shift of the original space. That is, the space shrinks in some areas and grows in others. I would expect this form of solution space change to be more common than the subset and superset ones, as it is rare that a mechanical change would be so localized.
5. The new solution space could be a disjoint set from the original. This outcome is the case where a mechanical change was so fundamental that I would argue it has created an entirely new game.

Determining which of these five relationships two solution spaces share is analogous to determining the level of co-occurrence between two sets of structured data. The intuition behind such an analysis is to produce something akin to a confusion matrix (like the example in Table 4), which I call an overlap matrix. For all the solutions present across two solution spaces, we want to know the number of solutions that appear in both spaces or uniquely in one space or the other. However, because it is non-trivial to define when two particular solutions are exactly the same, I conduct this analysis over representative solutions, similar to the solution space analysis presented in Chapter 6 , which is why the assignment of solutions to a space in Table 4 is not an integer.

	In A	**Not in A**
In B	2606.20 (68.24%)	1140.93 (29.88%)
Not in B	71.87 (1.88%)	**–**

A = 1907; B= 1912

Table 4. An example overlap matrix made by comparing the first two thirds (A) and last two thirds (B) of a subset of solutions from the formative evaluation of *RumbleBlocks*.

The process for calculating this overlap matrix is as follows. Given two solutions spaces A and B, I calculate three values: the proportion of each representative solution existing in both A and B (Overlap), the amount of each representative solution existing in A but not B (A Shift), and the amount of each representative solution existing in B but not A (B Shift). The first step of the process is to generate a set of representative solutions by taking all the solutions from both spaces and clustering them together using TRESTLE's [93] clustering procedure, optimizing for the Bayesian Information Criterion (BIC) heuristic (See Appendix B for details on the TRESTLE algorithm and its clustering process).

Given a set of clusters, the next step is to decide the proportion of each cluster that provides evidence of either an overlap or a shift. Because each cluster can be comprised of solutions from both spaces, a representative solution can have partial membership in each space. For example, clusters that have a 50-50 split of solutions from space A and space B would be viewed as sitting in the overlap between the spaces, where pure clusters (i.e., composed entirely of solutions from a single space) appear to exist in only one space. Clusters that are composed of some mix of the spaces could simultaneously suggest some amount of overlap and some amount of shift depending on the proportion of members and whichever one is dominant.

In calculating the percentage of a cluster that suggests overlap or shift, I use Binary Entropy [91], which measures the uncertainty of a binary classification. In effect, the more uncertain a cluster is about which space it contains, the more evidence that cluster provides for an overlap between the solution spaces. Entropy is used to weight the assignment of representative solutions to the quadrants of the overlap matrix. The Overlap quadrant is the sum of the size of the representative clusters weighted by their entropy. The A Shift quadrant is calculated as the sum of the size of the majority A clusters weighted by 1 minus their entropy and B Shift is calculated in a similar

manner for majority B clusters. Figure 12 shows a plot of how this weighting process plays out based on the percentage of solutions in a cluster that originally come from space A.

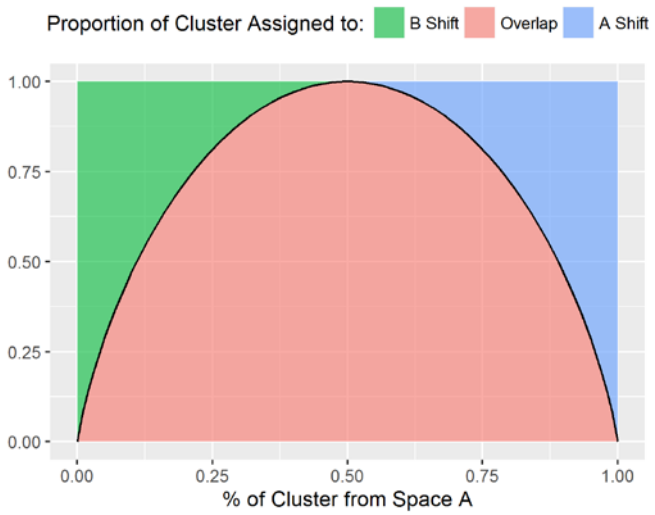


Figure 12. A plot showing how binary entropy (black line) is used to decide the proportion of each cluster that is assigned to the B Shift (green), Overlap (red), or A Shift (blue) quadrants based on the percentage of solutions in a cluster that come from Space A.

When comparing the whole space of two different game versions, I repeat this process on a level-by-level basis and sum the results of each level to obtain an overall comparison of the two spaces. For the purpose of whole space comparison, I drop any levels that do not have at least ten solutions existing in both versions. In the case where one version contains a level that the other does not, it is trivially obvious that any solutions to that level are either entirely new (in the case of an added level) or completely outlawed (in the case of removing a level) and so would bias the comparison.

Looking at the makeup of these overlap matrices allows me to categorize a pair of solution spaces into one of the five relationships I described above. If a large majority of solutions exist in the Overlap quadrant, then the solution spaces are effectively the same. If there is a large percentage of solutions in A Shift or B Shift with little to no solutions present in the other shift, then the solution spaces are in a sub/superset relationship with each other (e.g., I would characterize the matrix in Table 4 as space A being a subset of space B). If there is a spread of solutions across the three quadrants, then the spaces are shifting apart. Finally, if the overlap cluster is zero or nearly zero, then the solution spaces are disjoint. By employing this solution space overlap analysis on solution spaces generated by different Projective Replays, I can get a picture for how a game's space of possible solutions has changed under a new set of mechanics.

Projective Replays of Design Variations

In understanding exactly how each of the design variations I wanted to explore might affect players' experiences in *RumbleBlocks,* I performed Literal and Flexible Projective Replay analyses with each of the variations. Using the new solution space generated by the replays, I repeated the alignment regression (see Equation 1) from the formative evaluation and produced an overlap matrix comparing the new solution space to the one seen in the original data.

The close-the-loop evaluation study, which I describe in the next section, was performed using only 2nd and 3rd graders from one of the schools from the original formative evaluation. Given that this is the case, I make use of a sample of the original formative evaluation data, limited to just the 2nd and 3rd graders from the same school as the close-the-loop study, for the analyses in this section. This selection was done with the intent of reducing any possible noise due to population differences, under the intuition that the agents would be learning to emulate the players in the sample. If the agents were learning from the entire sample of data, it would be difficult to tell if any variations found when comparing to the close-the-loop study are due to changes to the game or variations in training data.

To generate a baseline solution space for comparison, I first performed a Retrospective Replay of the limited sample of the original game log data[17]. The resulting solution space is used as the baseline comparison space (referred to as Original) in all following space overlap analyses. I also performed a Flexible Projective Replay, trained on the limited sample, through the original game in order to see how the flexible projective agent would perform in the absence of any mechanical variations. The results of alignment regression over these two spaces can be seen in Table 5, and the overlap matrices from solution space comparison are in Table 6.

Replay Type	Coefficient	B	SE B	*β*	*p*	
Literal (n=2602)	(Intercept)	2.400	1.164	2.182	-	
	Base Width	0.137	0.068	0.188	0.043	*
	COM Height	0.053	0.131	0.039	0.686	
	Symmetry Angle	0.064	0.012	0.386	< 0.001	***
Flexible (n=1448)	(Intercept)	6.574	2.330	2.119	-	
	Base Width	-0.243	0.113	-0.342	0.031	*
	COM Height	0.391	0.251	0.242	0.119	
	Symmetry Angle	0.046	0.017	0.302	0.007	**

Table 5. Alignment Regression Results for Literal and Flexible Replays of 2nd and 3rd grade students in *RumbleBlocks*' original design.

Looking at the results of the alignment regression, we see that the experience from 2nd and 3rd graders in the original evaluation study was somewhat different from the overall population. Where the original alignment analysis highlighted a potential problem with the center of mass principle (see Table 2), the results for the restricted sample show more reasonable alignment for all metrics. The COM Height metric is still not significant, meaning its effect might be 0, but it is trending positive in this case, where it was negative before. The Flexible Replay results show a different pattern, where the Base Width metric is now more strongly associated with failure than success. This outcome suggests that the flexible agent would predict an altogether different alignment issue could happen with the game.

[17] I use replay here, as opposed to referencing the State field of the SAIS directly in the log data, because the clustering process requires a slightly different characterization of the state and to ensure that all data being compared were generated from a common *Interpreter* implemented in the RAE.

	In Original	Not in Original
In Literal Replay	5669.87 (98.95%)	30.18 (0.53%)
Not in Literal Replay	29.96 (0.52%)	–

Original = 2865; Literal Replay = 2865

	In Original	Not in Original
In Flexible Replay	3085.57 (81.50%)	162.95 (4.30%)
Not in Flexible Replay	537.49 (14.20%)	–

Original = 2343; Flexible Replay = 1443

Table 6. Overlap Matrices comparing solution spaces generated by Literal and Flexible Replays of 2nd and 3rd grade students with the original *RumbleBlocks* solution space. Note that the Literal matrix is the result of comparing the space to itself and is included for completeness.

Turning to the results of the solution space overlap, we see a pattern that is somewhat to be expected. First, in the comparison of Literal Replay with the original solution space, we see nearly 100% overlap as it is effectively the result of comparing a space with itself, because a Literal Replay on an unchanged game is essentially a retrospective replay as the game would treat the recorded player behavior the same[18]. When comparing the space generated by Flexible Replay with its the original counterpart, we see a substantial overlap (~80%) with most of the divergence existing in the Original space. This result would seem to suggest that the flexible agent generated a smaller space that is essentially a subset of the original space. This interpretation was confirmed when I looked at the percentage of solutions that the agent generated under its own reasoning versus in the process of taking a training example. Overall, ~10% of the flexible agent's solutions were generated under its own reasoning, while the remaining ~90% were the result of the agent taking a training example.

Once a baseline of behavior was established, I performed Literal and Flexible Replays of each of the design variations to be tested. The first variation I tested was adding the Glue Blocks mechanic to towers before the earthquake started. The alignment regression results for this variation can be found in Table 7 and the overlap matrices for the Literal and Flexible replays are in Table 8. In the alignment results, the Literal replay shows a negative trend for COM Height, while Symmetry Angle has a roughly similar effect to the original 2nd and 3rd grade data. This result would suggest that the Glue Block mechanic introduces misalignment problems similar to the current design as measured by the alignment of the overall data. Given that the rationale behind the Glue Block mechanic was to bring the game's mechanics more in line with the realities of earthquake physics, it is hard to understand why this might have happened. Looking at the flexible alignment results, we see a similar pattern to the Flexible Replay of the original version with the exception that none of the effects are significant.

[18] The reason overlap is not perfectly 100% is because TRESTLE is an order dependent algorithm and instances are fit into the tree in a random order, which can introduce minor noise.

Replay Type	Coefficient	B	SE B	*β*	*p*	
Literal (n=2612)	(Intercept)	0.462	1.329	3.283	-	
	Base Width	-0.166	0.087	-0.230	0.055	
	COM Height	-0.459	0.148	-0.341	0.002	**
	Symmetry Angle	0.082	0.016	0.509	< 0.001	***
Flexible (n=1327)	(Intercept)	6.231	3.076	3.369	-	
	Base Width	-0.232	0.161	-0.328	0.150	
	COM Height	0.232	0.331	0.147	0.484	
	Symmetry Angle	0.006	0.025	0.038	0.814	

Table 7. Alignment Regression Results for Literal and Flexible Replays of the Glue Block design variation.

Considering the solution space overlap analysis of the Glue Block variation (Table 8), we see that the Literal Replay again has substantial overlap with the original solution space. This outcome makes sense given that the Literal Replays contain the same towers with most of the variation being due to some cases where towers in the new condition failed to meet submission criteria due to idiosyncrasies in the newly implemented mechanics. The flexible overlap analysis shows more divergence than the original game version as the agent would now be learning from qualitatively different feedback due to the mechanical variation and thus likely would design more novel structures.

	In Original	**Not in Original**
In Literal Replay	5657.32 (98.73%)	34.23 (0.60%)
Not in Literal Replay	38.45 (0.67%)	–

Original = 2865; Literal Replay = 2865

	In Original	**Not in Original**
In Flexible Replay	2271.83 (61.99%)	394.71 (10.77%)
Not in Flexible Replay	998.46 (27.24%)	–

Original = 2343; Flexible Replay = 1322

Table 8. Overlap Matrices comparing solution spaces generated by Literal and Flexible Replays of the Glue Block design variation with the original *RumbleBlocks* solution space.

The next variation I tested was the No Cliff variation, which removed the physics collider of the cliff in the scene, feasibly opening the space for more variation. The alignment regression results of the No Cliff version are in Table 9 while the solution space overlap results are shown in Table 10. The alignment of the literal No Cliff replay is similar to the results of the Literal Glue Block replay but with a less pronounced effect for COM Height and instead a stronger relationship for Base Width. Again, COM Height and Base Width show negative associations with success on the level, suggesting possible issues with alignment in this variation. The Flexible Replay results closely mirror the Flexible Replay in the original version of the game. This outcome makes sense as the No Cliff variation is very similar to the original version of the game, with a slightly more permissive building environment.

Replay Type	Coefficient	B	SE B	β	p	
Literal (n=2626)	(Intercept)	1.134	1.045	1.707	-	
	Base Width	-0.176	0.070	-0.242	0.012	*
	COM Height	-0.194	0.117	-0.149	0.097	
	Symmetry Angle	0.079	0.012	0.475	< 0.001	***
Flexible (n=1411)	(Intercept)	5.492	2.310	2.003	-	
	Base Width	-0.140	0.114	-0.198	0.223	
	COM Height	0.320	0.250	0.195	0.202	
	Symmetry Angle	0.050	0.018	0.332	0.005	**

Table 9. Alignment Regression Results for Literal and Flexible Replay of the No Cliff design variation.

Again, in the solution space overlap analysis, the Literal Replay adheres very closely to the original solution space. The results of the Flexible Replay are similar to those of the Flexible Replay in the original version, with the new space being a subset of the original space. This result provides some additional weight to the interpretation that both the No Cliff and Original Flexible Replays had highly similar performance leading to similar alignment results.

	In Original	Not in Original
In Literal Replay	5614.80 (98.14%)	52.83 (0.92%)
Not in Literal Replay	53.37 (0.93%)	–

Original = 2865; Literal Replay = 2856

	In Original	Not in Original
In Flexible Replay	3049.48 (81.34%)	144.06 (3.84%)
Not in Flexible Replay	555.46 (14.82%)	–

Original = 2343; Flexible Replay = 1406

Table 10. Overlap Matrices comparing solution spaces generated by Literal and Flexible Replays of the No Cliff design variation with the original *RumbleBlocks* solution space.

The final variation I tested was the No Ship mechanic, which removed the influence of the spaceship on solution success and instead based success on the aggregate behavior of the tower. This variation was likely to see the most different behavior because the changes to the outcome mechanics were fundamentally different. In looking at the alignment results for the No Ship variation (Table 11), we see the now familiar pattern in the Literal Replay of a negative association with Base Width and a moderate positive for Symmetry Angle. The flexible results show a weaker negative trend for Base Width than most of the other Flexible Replays did and is still insignificant.

Replay Type	Coefficient	B	SE B	β	p	
Literal (n=2616)	(Intercept)	1.291	1.172	1.735	-	
	Base Width	-0.223	0.072	-0.305	0.002	**
	COM Height	-0.186	0.130	-0.139	0.153	
	Symmetry Angle	0.053	0.012	0.320	< 0.001	***
Flexible (n=1749)	(Intercept)	3.238	1.661	1.459	-	
	Base Width	-0.059	0.084	-0.079	0.480	
	COM Height	0.148	0.184	0.092	0.420	
	Symmetry Angle	0.063	0.014	0.379	< 0.0001	***

Table 11. Alignment Regression Results for Literal and Flexible Replay of the No Ship design variation.

In the solution space overlap results for the No Ship variation (Table 12) we see that Literal Replay again has nearly perfect overlap. Flexible Replay, on the other hand, shows another shift of solution space away from the original. This is likely because the No Ship mechanic presents fundamentally different feedback from the normal version of the game and, thus, a flexible agent is more likely to try things that have not been seen before, or be lead away from some solutions that used to be common.

	In Original	**Not in Original**
In Literal Replay	5625.32 (98.31%)	42.44 (0.74%)
Not in Literal Replay	54.24 (0.95%)	–

Original = 2865; Literal Replay = 2857

	In Original	**Not in Original**
In Flexible Replay	2328.35 (62.89%)	410.89 (11.10%)
Not in Flexible Replay	962.76 (26.01%)	–

Original = 2343; Flexible Replay = 1359

Table 12. Overlap Matrices comparing solution spaces generated by Literal and Flexible Replays of the No Ship design variation with the original *RumbleBlocks* solution space.

On the whole, these results show some differences in alignment between the original game version and each of the variations tested, generally favoring the original in terms of resulting alignment. The solution space overlap analyses tend to show a high degree of overlap with Literal Replay, as would generally be expected, and some divergence in Flexible Replay, depending on variation. Flexible Replay also tended to generate a partial subset of the original solution space, again depending on design variation.

Classroom Study of the No Ship Variation

The ultimate question of the classroom study is whether or not Projective Replay methods can be trusted as a means of evaluating an educational game on par with new playtesting. In order to address this question, I performed a close-the-loop validation of one of the design variations in order to see if the predictions made by Projective Replay are confirmed with a new population of human players. If there is agreement between the alignment and solution space overlap analyses with a population of real users, then I can be confident that Projective Replay Analysis is a valid replacement for new user studies.

This close-the-loop validation was implemented as a two-condition experiment, where one condition played the original version of *RumbleBlocks* while the other condition played one of the design variants tested in the Projective Replay. For this study, I chose to implement the No Ship variant because I felt it came the most clearly from my prior analyses and because the fundamental difference in feedback criteria had the potential for a broader space of player interpretation and thus exploration. Having the potential to explore more space would make it the toughest test of Projective Replay as the potential effects on the games' solution spaces are the hardest to predict *a priori*.

Students from two 2nd and two 3rd grade classes from one of the schools in the original formative evaluation were recruited. Students played the game during dedicated time in their classrooms using a WebGL build of *RumbleBlocks* on Chromebooks owned by the school. The study took place over four sessions. Similar to the original formative evaluation study, the first session was a pretest, followed by two days of gameplay, and finally a posttest. Unlike the formative evaluation, the pre- and posttests were administered digitally using a CTAT [6] based interface rather than with paper and pencil. Further, due to scheduling conflicts with the school, the posttest took place five days after the second day of gameplay, rather than on the next day, making it more of a delayed test. Overall, 77 students participated with full data. Participants were randomly assigned to a game version, with

40 in the base game condition and 37 playing the No Ship variation (see Appendix A for descriptive statistics from this study).

For this study, *RumbleBlocks* was altered from the original formative evaluation in a few additional ways. First, the original game contained a series of levels where players removed blocks with the goal of removing as many blocks as possible without toppling the tower. These levels were poorly explained in the context of the game and analysis of the log files showed that players spent an average of less than 5 seconds on these levels, suggesting that they did not engage with the material very deeply; thus, they were removed. Second, a follow-up study to the original formative evaluation expanded on the idea of contrasting cases as a game mechanic by introducing levels where players reasoned through why a set of towers stood or fell by looking through different magic goggles to visualize the underlying physical properties of towers. This study found a positive learning benefit of including these levels [29] and so they were included here in the interest of benefiting players. It is possible that these additional variations affected players' explorations of solution space during the study, in addition to the conditional variations of the mechanics.

When looking at learning effects on the external pre-posttests, as with the formative evaluation, there was a significant increase in performance on posttest using a repeated measures ANOVA $F(1, 68) = 12.10$, $p < 0.001$, but there was no significant effect due to game version $F(1, 68) = 0.88$, $p = 0.35$ and no significant interaction $F(1, 68) = 0.67$, $p = 0.41$. Looking at success rate on the in-game pre-posttests, in a repeated measures ANOVA, there was no significant difference in performance between pre- and posttest $F(1,74) = 0.09$, $p = 0.72$, no effect of condition $F(1, 74) = 0.02$, $p = 0.89$, and no interaction $F(1, 74) = 0.56$, $p = 0.46$. Repeated measures ANOVAs of players' metric use between pre- and posttest (Table 13) show a significant main effect for a greater application of symmetry $F(1, 49) = 12.31$, $p < 0.001$ from pre- to posttest across conditions, but no other significant effects or interactions.

Metric	Effect	MS	df	*F*	*p*	
Base Width	Time	0.38	1	1.01	0.319	
	Game Version	0.00	1	0.01	0.908	
	Time x Game Version	0.29	1	0.77	0.383	
COM Height	Time	0.41	1	1.44	0.232	
	Game Version	0.34	1	1.20	0.276	
	Time x Game Version	0.07	1	0.24	0.625	
Symmetry Angle	Time	2.99	1	12.31	< 0.001	***
	Game Version	0.16	1	0.67	0.418	
	Time x Game Version	0.56	1	2.32	0.134	

Table 13. Repeated Measures ANOVA results of players PRM use over time in the pre-posttest levels of *RumbleBlocks*.

The alignment regression results for both the Base Game and No Ship conditions can be found in Table 14. Looking at the Base Game condition, there is a non-significant negative association with COM Height. This coefficient shows a similar trend to the one seen in the original alignment analysis using the full formative evaluation sample (Table 2); however, this pattern does not match the one seen in the restricted sample of 2nd and 3rd graders in the original study (Table 5), suggesting there might be some variation here due to the different populations. In the No Ship condition, there are two significant effects for Symmetry Angle and Base Width; however, the effect of Base Width is particularly stark and negative, suggesting that there is a strong negative association with building wider based structures.

Game Version	Coefficient	B	SE B	β	p	
Base Game (n=1391)	(Intercept)	-1.109	1.669	1.232	-	
	Base Width	0.051	0.071	0.077	0.473	
	COM Height	-0.285	0.189	-0.182	0.132	
	Symmetry Angle	0.059	0.013	0.373	< 0.001	***
No Ship (n=1278)	(Intercept)	0.775	1.829	0.730	-	
	Base Width	-0.314	0.069	-0.473	< 0.001	***
	COM Height	-0.148	0.207	-0.094	0.476	
	Symmetry Angle	0.027	0.013	0.168	0.048	*

Table 14. Alignment Regression Results for the Base Game and No Ship conditions of the close-the-loop study.

For the solution space overlap analyses, I performed two sets of comparisons. First, I compared both of the new solution spaces to the original to see if there was a comparable amount of shift, as was predicted by the corresponding Flexible Replay. Second, I compared the new No Ship space to the Literal and Flexible Replays of the corresponding game version to see if these sets converged, suggesting that replay explored a similar space to what a new population of students would have done. The results of comparing the new spaces to the original can be seen in Table 15, while the comparisons of the new space to its counterparts are in Table 16.

	In Original	**Not in Original**
In Base Game	1850.96 (59.38%)	216.22 (6.94%)
Not in Base Game	1049.81 (33.68%)	–

Original = 2195; Base Game = 922

	In Original	**Not in Original**
In No Ship	1712.70 (58.02%)	169.59 (5.74%)
Not in No Ship	1069.71 (36.24%)	–

Original = 2116; No Ship = 836

Table 15. Overlap Matrices comparing solution spaces from the Base Game and No Ship conditions of the close-the-loop study with the original *RumbleBlocks* solution space.

When looking at the solution space comparisons between both conditions and the original solution space, it appears that both new spaces are mostly subsets of the original. On one hand, this pattern is to be expected because the new solution spaces are being created by a smaller number of players than the original and so it is more likely that there would be solutions the new players had not considered. However, it is surprising to see that there was not very much divergence in the No Ship condition, given that the mechanics of that condition differed so much from the original game.

	In New No Ship	**Not in New No Ship**
In Literal Replay	1682.90 (56.97%)	1083.60 (36.68%)
Not in Literal Replay	187.50 (6.35%)	–

Classroom = 836; Literal Replay = 2118

	In New No Ship	**Not in New No Ship**
In Flexible Replay	1009.45 (55.22%)	534.98 (29.27%)
Not in Flexible Replay	283.57 (15.51%)	–

Classroom = 748; Flexible Replay = 1080

Table 16. Overlap Marticies comparing solution spaces generated by Literal and Flexible Replays of the No Ship design variation with the solution space captured in the Classroom playtest of the No Ship condition.

The results in Table 16 show a similar trend as the rest of the close-the-loop comparisons where the replayed solution spaces are largely supersets of the spaces seen during the close-the-loop study. As would generally be expected, comparing the new No Ship space to a Literal Replay of the old solutions in a No Ship variation yields similar overlap results as when comparing it to the Original solution space itself. In the Flexible Replay case, there is a larger degree of divergence between the spaces, suggesting that the Flexible Replay into a No Ship variation does not strongly resemble what a new population of players would have done.

Discussion

The primary question asked in this chapter was: would a Projective Replay Analysis of a redesigned game come to the same conclusions as a full close-the-loop evaluation? To address this question, I performed two parallel evaluations of a design variant in *RumbleBlocks*. One evaluation was a Projective Replay Analysis, while the other was a close-the-loop study in the style of the original formative evaluation of the game. In repeating the alignment analyses from the formative evaluation, I expected to see similar relationships appear between success and PRMs in the results of evaluations of the same game version regardless of method of evaluation. Similarly, I expected to see a reasonably high level of overlap between solution spaces generated by players of the same game version and a similar level of shift away from the original solution space, whether those players were human students or AI models.

After performing the Projective Replay Analysis of the No Ship variation, the alignment regressions for literal projection show a significant positive effect from adherence to symmetry angle, and a strong negative association with base width. Flexible replay showed a similarly strong relationship for symmetry angle but did not show a strong negative effect for base width (the coefficient was negative but not significantly so). When looking at the results of the classroom evaluation, the alignment regression also showed a significant positive effect for symmetry angle and a strong negative relationship with base width.

Looking at the results of solution space overlap analysis, the Flexible Replay of the No Ship variation shows some shift away from the original solutions (~11%) but for the most part is a bit of a subset of the original space with ~26% of the original space not seen in the Flexible Replay. When looking at the level of shift from original in the classroom playtest, the No Ship condition looks like a subset of the original space. Only ~6% of the joint solution space exists only in the new data with ~36% appearing only in the original. When looking at the relationship between the two No Ship spaces, the pattern looks like a diverging shift with only ~55% of the solutions appearing in overlap and the rest split roughly two-to-one between the replayed space and the human student space respectively.

Given the preponderance of evidence from the close-the-loop study, it is difficult to decisively conclude that Projective Replay, as applied to these game variations and implemented in this study, was a fair substitute for a real classroom study. The alignment conclusion arising from the classroom study is generally that the Base Game condition would continue to have some issues with its treatment of the Center of Mass principle while the No Ship condition will manifest a strong negative association with Base Width. When comparing this pattern to the one found in the Projective Replay of the No Ship variant, the Literal Projective Replay does predict this relationship, however the Flexible Projective Replay does not. This result would suggest that Literal Projective Replay can predict changes in alignment where Flexible Replay did not. Given that Literal Replay is essentially a naïve implementation of a replay agent, it is disappointing to see it perform so well relative to a presumably more intelligent flexible agent.

Considering these results, the next question is: why might it be the case that a naïve agent implementation would outperform one that is capable of learning from new feedback and potentially better at exploring new game spaces? One possible explanation is that the agent, as implemented, was not able to generalize the demonstrations it received well enough to learn suitable skills. Potential support for this hypothesis can be seen in looking at the percentage of solutions the agents created while in a trying orientation as opposed to watching. Of the 1749 solutions in the flexible agent's solution space, only 151 (~9%) were produced in a trying orientation. This pattern would explain why the flexible agent's solution space appears to be mostly a subset of the original space, as almost 90% of its solutions are the result of the agent watching a prior logged solution. If the agent was able to properly generalize demonstrations, I would have expected to see a much larger portion of solutions produced from a trying orientation.

One hypothesis I have as to why this happened deals with the impasse points that I mentioned in the implementation of the agent's training protocol (see the Flexible Projective Replay Agent section pg 22). Under the current training paradigm, if the agent has produced several actions under its own reasoning but then finds it has no applicable skills it will request a new demonstration from the logs. In such a case, the next logged action will not be applicable to the current situation, as it has fallen behind the agent's own reasoning. Currently, in these situations, the agent merely forgets the experience and is moved on by the system. It is possible that other methods of providing demonstrations could be employed to allow agents to continue to explore new paths more fluidly without backtracking in dead ends. For example, if an agent finds itself ahead of its log file and out of ideas, a secondary demonstration source could be used to find a demonstration across all logs that would be applicable in the current situation (e.g., some existing work with Monte-Carlo tree search [172] could serve this purpose) whether or not it came from the same student or the same level. Alternatively, more general AI approaches could be used to allow the agent to plan its way out of impasse problems.

Another possibility as to why I saw primarily a subset relationship from the flexible agent is that this use case was arguably stacked against Flexible Replay. Going into the evaluation, I chose the No Ship condition because I had no strong *a priori* intuition about how the mechanical change would affect the solution space. On a fundamental level, no core interaction mechanics were changed so it was entirely possible for the solution space to remain the same, as all prior towers could be validly constructed again. If there was a mechanical change that made it impossible for Literal Replay to work in new space (e.g., introducing a new concept like a negative energy dot that destroys blocks when touched), then the Flexible Replay may have provided a better picture of how the new game could work because it could potentially learn from these new designs. However, given my hypothesis with regard to generalization of examples, we might have still seen a similar problem.

More generally, it is hard to tell how much of a new solution space could have been expected. The original premise behind my solution space analysis was that *RumbleBlocks*, or any game for that matter, has as solution space of some unknown size. Using more formal representations for a game, it might have been possible to map this space *a priori* [134,147]; but operating outside of those assumptions, it is less clear how large the fundamental space is. It is possible that the particular affordances of blocks and energy dots within *RumbleBlocks* do not allow for substantially more variation than had already been seen in the formative evaluation study. A limitation of trying to measure solution space overlap is that it is hard to know how much overlap would be expected due to chance. While the overlap matrices show overlap and shift among their three quadrants, in a more objective frame there could exist a non-zero quantity in the fourth quadrant, namely a space of solutions that do not exist in either space A or space B due merely to sampling limitations rather than fundamental mechanical implications. It is possible that the data captured in the original formative evaluation was a sufficient sample to see all of the

reasonable variations that 2nd and 3rd grade students would try in the game without fundamentally changing interaction mechanics. In such a situation, Flexible Replay would, at best, be expected to produce an overlap if not a subset, as we saw here.

The second major question of this chapter was to understand if following the recommendations of an alignment-focused analytics process can produce a better aligned game. To answer this question, I ran several more Projective Replays of different design variations arising from my analysis to see if the prediction trends of PRMs related to success would improve after making a change. Ideally, we would see from these replays that the alignment regressions of the changed versions of the game would show better relationships between the PRMs and success than the original formative evaluation of the game.

When looking at the results of the regressions, we see some interesting patterns. First, in the Literal Replay of the Glue Block variation the negative coefficient for COM Height that existed in the original formative evaluation as a non-significant effect has become significant and stronger, meaning that the Glue Block condition may actually be worse aligned with regard to the COM principle. As was mentioned previously, both the Literal Projective Replay and the human playtest of the No Ship condition predict a new strong negative relationship with base width that was not present in the original data. The No Cliff variation also shows this negative effect for base width. Perhaps most curious is the fact that the new classroom evaluation of the original game version does not agree with the alignment conclusion of the 2nd and 3rd grade subset of the original data, as the new data does not have a significant relationship between base width and success when the original data did. The general trend across these results is that no new variation is predicted to be categorically better than the original data. Most of the alignment results predict a positive relationship for the symmetry principle, but beyond that each variation has different strengths and weaknesses.

One possible explanation for the strong negative association with base width in the No Ship condition comes from my observations of students playing the game during the study. The scoring function of the No Ship mechanic introduced an unforeseen dynamic that affected some towers. In several levels of the game, players stacked blocks in wide flat structures covering several energy dots (See Figure 13). When the earthquake hit those towers, they tended to jump up momentarily and lose contact with multiple energy dots at once. This momentary lapse resulted in a strong negative hit to a player's score, often resulting in them losing the level because scoring in the No Ship condition is based on average number of energy dots covered over the frames of the earthquake. This idiosyncrasy frustrated several players in the classroom playtest and is the kind of unintended consequence that a playtest would want to discover.

Figure 13. An example of a tower that fails in the No Ship condition because of the unforseen dynamic of the scoring function. The brief moment where three energy dots are uncovered (middle) causes the tower to fail.

It appears that Literal Projective Replay could have predicted the negative consequence of the No Ship design variation given the comparable patterns that arose in the alignment analysis. The reliance on classroom

observation to inform the conclusion about the mechanical problem is primarily due to time constraints. I would fully expect a further solution space analysis of the new classroom data to highlight the problems with this design pattern and I plan to explore this in future work.

It is interesting to see that the alignment conclusion of the base game condition in the new study does not agree with the original study. One possible explanation for this is a sample size issue, given that the base width relationship in the original data is just on the acceptable side of significant. Another possibility, however, could be implementation differences between the two playtests. The original playtest took place on locally installed versions of the game, while the new playtest was performed on a WebGL build of the game. As Unity's physics engine is hardware dependent, there is a possibility that it would behave somewhat differently across different platforms and introduce noise. The Projective Replay process could actually be used to run replays of old sessions on new hardware and directly measure this effect, but I did not have the opportunity to test that in this instance.

While I was unable to find support for the conclusion that alignment focused redesign ideas generate games with better alignment, that does not mean the hypothesis is wrong. It is certainly possible that the solutions I proposed to alignment problems in *RumbleBlocks* were merely poor answers to a valid problem. From an iterative design perspective, these Projective Replays would be used as evidence to eliminate the current set of redesign candidates, but other good ideas may still exist. Doing these kinds of validations and closing the loop is actually a strength afforded by Projective Replay, as performing full close-the-loop style playtests for each of these variations would have been prohibitively expensive in time and social capital. Endeavoring to go beyond the tests I performed here to find a better design solution to the alignment problems I observed could be a potential avenue for future work. Using a broader exploration of this process with a practicing design team that would not be influenced by my own preconceptions of *RumbleBlocks* might uncover an even better answer to the design issues of the game.

CHAPTER 8 IMPLICATIONS, CONTRIBUTIONS, AND CONCLUSIONS

In this thesis, I have endeavored to demonstrate how replay-based approaches can aid iterative educational game design with an eye toward pedagogical alignment. Within the body of this work, I have presented a new formulation of pedagogical alignment as a relationship between principle-relevant metrics and game feedback and developed a method for measuring this relationship using log data. I have also described a taxonomy of Replay Analysis methods and developed a toolkit for implementing replay within the Unity game engine. I demonstrated the utility of Retrospective Replay in a formative evaluation of the alignment of the game *RumbleBlocks*. I further showed how retrospective replay could be used to understand a game's solution space from multiple perspectives. Finally, I explored the possibility of using Projective Replay Analysis to evaluate a game design iteration without performing additional playtesting.

Given the results of all of this work, I will now discuss what can be said about Replay Analysis and alignment and their place within the iterative design of educational games. I will separate this discussion into two parts, one for each of the core questions, before detailing what I see as the contributions of this thesis and discussing implications for future work and conclusions.

Discussion of Replay Analysis

In presenting Replay Analysis in its various forms as part of this thesis, I sought to demonstrate that it is both a useful and valid approach to investigating potential problems with an educational game's design. I believe I have succeeded in demonstrating the utility of the method through my work and have provided some qualified support for its validity. In this section, I will discuss how I arrived at this conclusion.

It could be argued that none of the analyses I performed in Chapter 5 or Chapter 6 using Retrospective Replay Analysis required that I use a replay paradigm, and I do not disagree with this statement. I do not intend to claim that Replay Analysis uniquely enables any one kind of investigation into a given game design; instead, I argue that it affords the ability to adapt and reframe data as needed by an evolving investigation. One point in favor of this conclusion is that every analysis within this thesis, with the exception of the external pre-posttest comparisons, was facilitated in some way by a Replay Analysis approach and, in most cases, using a single dataset.

Using Replay Analysis, the data from a single classroom playtest of *RumbleBlocks* was able to provide insight into:

- whether players appear to be learning target concepts and can apply them better after playing
- whether the game provided feedback to players' solutions in a way that aligned with its goals
- how players approach solving levels differently and how their approaches relate to designers' expectations
- how the structural patterns in players' solutions predict how the game will react to them
- how the game's behavior might change after altering its mechanics in three different ways

Whether any one of these analyses possess utility is a subjective question to a particular design situation, but the ability to freely move between them is where I see the value in Replay Analysis.

Another way of looking at Replay Analysis is as a form of deferred analytics. It is trivial to envision an implementation of a record-to-measure analytics system that would be sufficient to perform any one of the investigations in this document in isolation. Such a system would look similar to the various *Interpreter* components I implemented for the RAE to perform the analysis myself. The main difference is that, rather than spending time developing an entire analysis plan before designing the game and tasking a game developer with

implementing several specialized telemetry hooks, I was able to create new *Interpreters* for the data on the fly as my questions about the game design evolved.

With regard to the validity of Projective Replay as a game evaluation technique, I find the results mixed. When applying Literal Projective Replay to the No Ship variation of *RumbleBlocks,* the alignment regression results showed a strong negative association between a wider base and success. This same pattern was found in the alignment regression results in the close-the-loop classroom study of the No Ship variation. Both of these patterns are likely caused by the mechanical issues I observed while running the playtest in classrooms, where certain tower designs with wide flat structures were adversely affected by the new scoring function. The fact that a Projective Replay was also able to show this pattern without new player data would suggest that it possesses some validity as method to allow designers to explore mechanical ideas more quickly. In this case, Literal Projective Replay is functioning more like a regression test [106] to show that a change to the game created adverse side effects. While it was hoped that a Projective Replay would be able to validate an improvement to the game, being able to catch a new fault is equally important to the design process.

The question of Flexible Projective Replay's validity is more complicated. Across all of the Flexible Projective Replays I presented, the general conclusion is that they each explored a subset of the space of the original game and little beyond it. I interpret this to mean that the flexible projective agent, as currently implemented, is not able to adequately generalize from prior players' log data. Fully explaining why this is the case demands future work, but I have several hypotheses that relate to the agent's design.

First, the way the agent currently handles impasses between generating actions and requiring demonstrations (e.g., being unable to reason about an action while several steps ahead of its example trace or ending up in a cycle of non-operative actions) is to forget experience and reset itself. By dropping experience, the agent is missing out on learning potentially useful information. A more intelligent impasse resolution approach could allow the agent to continue longer in exploring the space on its own without relying so heavily on the particular example trace. For example, the agent system could be augmented to try looking for the most applicable demonstration from across an entire dataset when it gets too far ahead of its example trace. One issue with this solution is that it would move away from the intention of the agent system to be a model of a single player's behavior if it could take demonstrations from beyond one player's experience.

Alternatively, the agent system could take an approach of interactive training by asking a designer for a demonstration when it is out of actions, similar to the implementation of SimStudent for authoring expert models in tutoring [95]. This strategy would certainly provide a means for the agent to resolve impasse situations and likely better prepare it for game variations that differ wildly from the original game, but it does have a few downsides of its own. Currently, the Flexible Replay process takes a significant amount of time, often running over multiple days. If an agent was implemented to require interactive training from a designer, they would need to be able to provide a demonstration on demand while the process ran, or employ some kind of queueing strategy. Further, having designers train their playtesting AIs by hand runs the risk of introducing expert blind spots [78] in the evaluation where the agent is really learning to emulate the thought processes and assumptions of the designer and not the behavior of a novice learner who may have a different perspective on the task. In so far as playtesting is done to learn something about a product that its designers do not already know, having an agent play from a designer's perspective is less likely to uncover new information.

A second possible limitation in the agent's implementation comes from how feedback is assigned to actions during the learning process. Currently, the agent assumes it will be given feedback at the end of a series of actions, and

when learning from the experience the feedback is assigned equally to each action. This strategy could be seen as a naïve approach to the credit/blame assignment problem, though it has some precedent in AI literature [83]. The essential issue with the credit assignment problem is that not all actions that a player would take in attempting a level cause it to be a success or failure. An example from *RumbleBlocks* would be that placing the first block on the ground does not necessarily cause the entire tower to fail; however, that is how the current implementation of the agent would see it. More sophisticated approaches exist in the literature on reinforcement learning [104], but these approaches are commonly aimed solely at reaching competency from experience, not in following a learning trajectory similar to a human learner. My implementation adheres more to the goals of the Apprentice Learner Architecture to model and explain human learning not only in asymptotic output but also in trajectory [94]. The apprentice learner paradigm currently bypasses this credit/blame assignment problem because it was originally designed within the context of intelligent tutoring systems and thus assumes a pattern of display-based reasoning [84], where actions are taken at a fine grain size and feedback is provided immediately [159]. Exploring ways to augment the architecture to better account for the credit assignment problem will be a component of my future work.

A further way that the agent could be improved would be to change the internal *how*, *where*, and *when learning* mechanisms. One potentially promising example, given the lack of generalization in the agent, would be to use a general-to-specific *where learner* instead of the specific-to-general one that is currently used. The Apprentice Learner Architecture was intentionally designed to be modular in this way and allow for different internal algorithms to represent the various learning mechanisms. The particular model that I use here was chosen because a similar version without TRESTLE had demonstrated some capacity to model human learners in the past [94] and adding TRESTLE was likely to create a model better suited to *RumbleBlocks'* highly structured environment. The infrastructure did exist for me to explore varying the internal implementation of the agent, but it was out of scope for this thesis[19]. The ultimate goal of the Apprentice Learner Architecture is to find some set of mechanisms capable of modeling learners' behaviors across a number of domains simultaneously and further iterating on the implementation of the flexible projective agents will be part of this future work.

An additional question raised by my discussion of Replay Analysis is the degree to which the approach would generalize to other games and other systems as well as where it would fit within the process of game design more generally. All the work I have presented as part of this thesis has been in the context of a single game. While Replay Analysis was applied in a more limited capacity to some of the other games developed as part of the ENGAGE project, most notably *Beanstalk* [31,58], it remains an empirical question how well the approach would generalize to other games.

Throughout the development of the Replay Analysis toolkit, I took care to avoid tailoring the system too heavily to the particulars of *RumbleBlocks'* implementation. The RAE is implemented as a completely separate software package and anything that needs special knowledge of *RumbleBlocks* (e.g., *Interpreters* and the *GSRC* for *RumbleBlocks*) are isolated from the main library. Further, all of the core components of the RAE are designed to mimic software patterns present throughout Unity's standard library to make integration with new systems easier. While these intentions are good, a proper validation of developer ease of use in integrating the system into a new game would be a topic for future work.

[19] As per my previous footnote, this application of the Apprentice Learner Architecture is, in fact, the topic of Christopher MacLellan's thesis, which will be published shortly after this one.

One question with regard to generality relates to how actions are encoded and what assumptions this might impose on a design. The SAIS structure theoretically has the capacity to encode any action that could be taken in a game, but because the description is contextual rather than based on raw input, like a mouse click, there is a question of what is an appropriate level of description for an action or step [160]. DataShop's original SAI encoding is commonly used to describe semantic actions applied in the graphical user interfaces of intelligent tutoring systems and has sufficed for a wide range of interfaces as evidenced by the volume of data contained within DataShop itself [75]. One of the reasons this has been so successful is because the interaction paradigms afforded by most graphical user interfaces are reasonably consistent. The space of games, on the other hand, is populated by a much larger set of possible interactions that players could apply to entities in a game world; and, while it may be theoretically possible to encode any of them in a SAIS structure, it is not trivially obvious that it would be easy without some sufficiently clever strategy. An example that SAIS would have trouble encoding is a continuous action, such as navigating a 3D space. This limitation is why I restrict my focus to step-based interaction paradigms, making a similar assumption to DataShop. In so far as a game is also step-based, as many educational and puzzle games are, my methods should transfer readily or with minimal difficulty. In other cases, some kind of extra encoding would be required. For example, by leveraging the concept of functional game space [135], one could impose an encoding where players navigate between meaningful points of interest in a 3D space as discrete steps rather than over continuous paths in meaningless space. Exploring how to better understand players interactions in more open-world environments is a topic I am interested in exploring further.

Another possible limitation on generality imposed by encoding is how the attempt, level, session, and user description of action progression makes assumptions about how a game is structured. Currently, the system assumes a progression pattern like the one in *RumbleBlocks*, where players progress through a series of levels contained within a larger hierarchy of tiers. While this assumption is limiting, it is also an extremely common structure in games from *Mario Brothers* to *Angry Birds*. It would be hard to apply this encoding to games that do not have a discrete set of challenges, such as *Minecraft*, though such games could be coerced into the format by imposing some discrete attempt structure onto in-game tasks. Further, there would be similar limitations for games that have procedurally generated or randomized level designs where the ability to call up a scene by name cannot be guaranteed. Such a game would require some additional encoding in the replay to ensure things like a randomizing seed or other invariant properties are captured in state descriptions. It is not necessarily required that the system be specifically designed in this way, though changing some of these assumptions would impact how the agent's interactive training process, such as when feedback would be provided.

A final structural limitation built into the Replay Analysis Toolkit as implemented is a focus on single-player experiences. The system currently assumes that actions are recorded from a single user's perspective and that they should be played back as a single agent. It is not impossible that the replay process could be extended to a multiplayer context as the software systems for enabling real-time multiplayer have much in common with replay systems [44]. Adding multiplayer functionality to the system would require substantial additions to the agent framework, and the Apprentice Learner Architecture itself and so it must remain future work for the time being.

Ultimately, I regard my efforts in presenting Replay Analysis to have been successful in demonstrating the utility of the method to allow for a wide ranging and reflective analysis process. I have also shown qualified evidence for its validity as a rapid prototyping method on part with conventional playtesting, but further work is needed to explore the promise of the method in more contexts.

Discussion of Alignment

My goal in presenting a new framing of alignment as part of this thesis was to demonstrate that looking at alignment as an agreement between a game's feedback and Principle-Relevant Metrics provides some much-needed grounding to discussions of alignment in educational game design. As with the discussion of Replay Analysis, my work on alignment also requires a discussion of the utility and validity of the approach toward aiding the educational game design process, considering the evidence of my work. I find the results with regard to validity to be generally successful while the results for utility are more qualified.

Regarding the validity of my formulation of alignment, the results of both studies present some encouraging evidence. The in-game pre-posttest results of the formative evaluation showed a significant learning effect on the Base Width and Symmetry Angle metrics, but no significant result for COM Height. The alignment regression results from the same study show a similar pattern, where the game showed appropriate feedback for Base Width and Symmetry Angle but not for COM Height. In the second classroom study, I see a similarly encouraging result. In the pre-posttest results, repeated-measures ANOVA results showed significant learning of the Symmetry Angle metric across conditions. This pattern corresponds with the alignment regression results where both conditions showed a significant positive relationship with Symmetry Angle. Taken together these results show encouraging support for my formulation of alignment as regression. In addition to providing some face validity to this approach these patterns also demonstrate some correspondence to the findings of Cohen [36] that a better aligned instructional program leads to greater learning.

While the correspondence of alignment and learning does provide encouraging support for my method there is still an interesting pattern in my second classroom study that is left unexplained. The alignment regression results of the No Ship condition show a strong negative effect for the Base Width metric but the pre-posttest learning results for that condition did not show a corresponding negative learning trend. As I previously discussed in the alignment chapter, whenever a game is rewarding unprincipled behavior or punishing principle behavior (the pink off-diagonal of the matric in Figure 1) then at best the game would confuse players and at worst it would foster misconceptions. The pattern shown by the No Ship condition appears to be the former case of merely confusing players. This is obviously the more desirable of the two possibilities but does raise interesting questions for future work on what conditions are necessary for misalignment to foster an active misconception.

A different angle on the question of the validity and utility of my alignment framing stems from its contribution to the iterative design process. If my approach finds a misalignment issue with a game and a variation to the design is made to correct that issue, we would expect to see greater alignment as a result. When looking at the results from the close-the-loop evaluation, this hypothesized outcome is not what occurred. The exploration into *RumbleBlocks'* solution space suggested a possible cause of the misalignment problem in the outsized effect of the spaceship on a player's success. In an effort to correct this issue, I explored three different variations of the game and measured their potential change in alignment with Projective Replay. Accepting some uncertainty due to potential limitations of Projective Replay, I still did not see an improvement in alignment in any of the variations. It is tempting to consider this result an indictment of my alignment approach, but it is possible that a decrease in alignment would have happened anyway, suggesting that alignment is hard to predict. The design process is wicked [24] and unintended implications of design choices are common. In this light, I view my perspective on alignment as a useful mental model [48] for thinking about the design process rather than an objective function to optimize a design.

Similar to the discussion of Replay Analysis there are also questions of generalizability for my formulation of alignment. In particular, what assumptions underlie the definition of PRMs and how might that restrict the kinds

of games that the technique could be applied to? Viewing alignment as a regression task provides some perspective on this question. Through this lens, a PRM could be anything that should predict success. In my case PRMs were relatively straightforward numerical properties of game state but other approaches are possible. For example, one could imagine a different *RumbleBlocks* with the goal of teaching that arch-shaped structures are more stable. In this case a categorical PRM could be defined to label towers as *is-arch*, rather than define some abstract numerical metric for *arch-ness*. In the regression formalism, this *is-arch* metric would be treated as a categorical factor and would result in an estimate of the effect of following the arch principle. Alternatively, the outcome side of the regression could be continuous rather than binary necessitating a different form of regression (e.g., Gaussian or Poisson) but ultimately functioning similarly.

The simplistic framing of alignment as measuring PRMs that should predict success does open up a question why stop there? It would be tempting to try and invent as many PRMs as possible and try to fit them all into a massive omnibus regression. This strategy would run into issues of statistical power and limit the technique's ability to accurately diagnose issues with a design. Fully addressing all the ways one might address this problem is well beyond the scope of this thesis and is more the topic of focused discussions on statistical methods [141]. I would, however, suggest some rules of thumb for practitioners looking to replicate my approach. First, it is better to focus alignment measurement on particular metrics of interest, likely agreed upon with subject matter experts or stakeholders, rather than try everything. Further, it is difficult to say precisely how much data is needed for the method to be accurate, beyond saying more is better, as this depends upon the shape of the regression formula being applied. This problem is aided somewhat by the unit of analysis for alignment regression being a player solution rather than a player themselves, which allows a single playtest to provide more signal about a game than one might expect. Ultimately, the regression-based alignment approach does require some sophisticated statistical training to apply correctly, and exploring ways to reduce this requirement would be interesting to explore in the future.

Overall, my conclusion with regard to measuring alignment as a relationship between in-game metrics and feedback shows promise as a metric for evaluation a design but more work is needed to better understand its bounds. The results from comparing alignment measurement with learning gains show encouraging potential for positive alignment to predict positive learning, however misalignment does not necessarily predict a misconception. Further, designing toward alignment did not produce a better aligned game in my case, however it is still useful to understand when well-intentioned design choices have unintended consequences. As it stands I would not advocate for using alignment as an exclusive objective function to drive the design process, but rather hold it as one measure among many to help designers think about what their designs look like and how they might move forward.

Contributions

Collectively my work makes contributions to the fields of educational game design, learning science, and human-computer interaction.

My work advances the field of educational game design by contributing a number of novel methods for helping designers to understand their games. Retrospective Replay Analysis provides a novel method for designers to characterize playtest data from a number of perspectives and suit varied analyses, expanding on prior playtesting methods by making individual playtests more valuable. Alignment Regression formalizes the relationship between game goals and game behavior, allowing for a common language to talk about expectations in a design. Solution Space Analysis opens a window into the varied experiences that players can have in a way that is digestible to designers. Finally, though more work is needed, Projective Replay shows promise for allowing designers to predict

the implications of their redesign choices. Taken together these methods form the beginnings of a toolbox for educational game design that can be expanded in the future.

In the learning sciences, my focus on measurements of the alignment of instructional moves to goals stands to benefit the literature on alignment research, which is typically more focused on the alignment of assessments to goals. In particular, my definition of alignment in terms of a solution space is a novel way to approach the alignment of instructional moves and provides an empirical basis for attacking the question. Further, the characterization of alignment as a regression task gives a well-formed answer to the question of what it means for elements of an instructional context to agree with each other. More work is needed, however, to fully understand how alignment from this perspective relates to student learning.

In human-computer interaction, replay paradigms have been an established means for the evaluation of interactive software. In exploring Projective Replay Analysis, I am expanding on this prior work by probing the boundaries of how the nature of design changes affect the validity of replay based methodologies. The results of my studies provide some encouraging support for the use of a Literal Projective Replay approach to evaluate next game design iterations. Further, my exploration of Flexible Projective Replay shows promise for expanding the paradigm in the future to allow for AI-enabled evaluation of future game designs in the absence of new player data.

Implications for Future Work

Beyond the contributions of my work for the originally intended purposes, the results hold interesting implications for exploring new areas of research in the future.

From Implications for Design to Implication of Design

While developing the method to characterize how a design change affects a game's solution space, an implication arose for further studying the relationship between a designer's choices and their implications on experience. Currently, I frame my work in terms of developing a tool to aid a designer in thinking about a particular product in a particular context, for a particular purpose. I heartily believe in this goal of aiding design at a hyper-local level, but there is an opportunity in this work to explore a broader set of questions.

The broader perspective of this concept is that design decisions have side effects. This notion is not a new one [9], but prior work has lacked a consistent way to characterize these side effects. By looking at ways to characterize the relationship between the behavior of a design before and after a change is made, we can start to explore what other implications of a given choice might reliably arise across designs. An example where I see potential here is in considering the space of learning science design principles [76]. In general, learning science design principles are framed as a claim that following a certain instructional form (e.g., interleaving topics instead of presenting them in blocks) is demonstrably better for learning. In essence, these principles are design patterns for good instructional practice [10]. What is less often talked about with these principles is whether any side effects might result from adhering to them. As I have discussed several times in this thesis, design is a wicked problem [24] and while recommending a given principle is likely to be beneficial, that abstract recommendation may be eliding design details that are important to a specific context. Investigating if instructional design principles possess reliable side effects could provide more nuance to discussions of instructional design.

Material Experience

An implication of the replay paradigm that I find particularly interesting is what it suggests for the space of experience design. Literature on experience design often casts designing an experience as explicitly different from

designing a material object [63]. An experience is seen as something that is subjective, holistic, situated, and dynamic [64]. These qualities are also likely what lead game designers to characterize their practice as second order design [133]. When the target of design is so subjective, prototyping design ideas becomes challenging [25] because so much of the experience depends upon a users' own reactions and understandings. In these situations, Schön's notions of seeing [138] become difficult without extensive experiential testing.

Operating under a replay paradigm offers the chance to reify experience into a material form. The situated and dynamic nature of experience can be re-instantiated in the context of a replay and considered from a design lens. In furthering the development of my replay approach and its associated software systems, I am interested in further exploring how this implication changes the design process and the relationship between designer and game (or product more generally). If, in the midst of exploring an idea, a designer could summon the opinions of an army of virtual playtesters, how would this change their iterative process? Further, how does building such an army change over the life cycle of design? Could a single small playtest act as a seed for growing more playtesting agents as iteration continues?

One aspect of experience that would elude replay is the appreciation of a player's subjective experience. As a designer, perceiving a player's experience from the outside will always have some limits on truly appreciating how a player feels about it. Existing models of player experience often rely on survey measures or interview techniques to probe how players feel about a design [43,107]. However, there is an opportunity to expand upon the player modeling approach of the Apprentice Learner Architecture to include processes for emotion and affect. Existing work on integrating affect into models of learners and cognitive systems [132,167] could be a fruitful direction to explore and will be a component of my future work.

Conclusion

In this thesis, I have presented the method of Replay Analysis as a technique that uses in-game replays as a data source for varied analyses of educational games. I created the Replay Analysis toolkit for the Unity game engine that embodies this method. I applied Replay Analysis to the educational game *RumbleBlocks* and demonstrated its capacity to support varied analyses. Using Retrospective Replay Analysis, I developed a novel framing of the instructional alignment between a game's educational goals and its instructional moves. I further employed replay to re-characterize the game's solution space from several perspectives in trying to diagnose why it may have alignment problems. Finally, I explore a new use of replay in projecting old recordings into new iterations of a game design and evaluated whether such a method would reach valid conclusions about a game on par with conventional playtesting.

The results of this work have shown promise for both Replay Analysis and my characterization of alignment. In exploring Projective Replay, I found that an exact reproduction of a recorded game session was capable of predicting some issues with misalignment in a new game version, while an AI-augmented approach was less effective. This suggests some potential future work in expanding the AI model to enable further generalization from training examples. In validating my formulation of alignment, I found mixed results. While in one study there was a strong agreement between a pattern of misalignment and learning, in another there was a weaker association. Further work is needed in this area to understand how other contextual factors of instruction contribute to student's learning or lack thereof.

As I stand at the culmination of my doctoral studies, I find myself at a place that looks more like a beginning than an end. The work I have done so far in developing a new method and toolkit has created a useful first step to explore several future ventures. In continuing to develop both Replay Analysis and alignment measurement, I

hope to continue expanding our understanding of educational game design, learning science, and human-computer interaction.

APPENDICES

Appendix A. DATASET DESCRIPTIVE STATISTICS

Throughout this appendix I have included some general descriptive statistics of the datasets used throughout my various studies. Overall, the purpose of this appendix is to provide context; I make no inferences or conclusions from the data presented here. However, one of the reasons I included this appendix was to show the distributions of the PRMs used in alignment regression, as they are somewhat irregular.

Formative Evaluation Study

The data in this section is from the formative evaluation study presented in Chapter 5 . The data in Table 17 is presented organized by school and grade. School A is the school that was used in the follow up classroom study presented in Chapter 7 .

School	Grade	Total Students	Total Attempts	Avg Levels Completed	Std Levels Completed	% All Attempts Successful	% First Attempts Successful
	k	37	847	19.43	7.01	81.46	82.51
	1	36	1139	24.78	7.76	80.42	81.32
A	**2**	39	1356	27.10	7.71	81.49	83.35
	3	39	1492	29.62	6.48	83.31	83.58
	Overall	151	4834	25.32	8.11	81.80	82.79
	k	27	591	17.67	6.69	75.13	77.45
	1	45	1186	21.04	6.39	75.04	76.44
B	**2**	30	863	24.03	6.99	78.91	80.06
	3	28	997	29.54	5.65	80.04	82.80
	Overall	130	3637	22.86	7.58	77.34	79.24
Overall		281	8471	24.18	7.95	79.88	81.23

Table 17. Descriptive statistics of the user population for the formative evaluation study of *RumbleBlocks*.

Figure 14, Figure 15, and Figure 16 show density distributions of the three principle-relevant metrics used in alignment regression. These plots show the raw scores for each metric across all of the data. Note that while these distributions appear very non-normal, this effect is minimized in the regression because of the inclusion of a random intercept for level (see Equation 1).

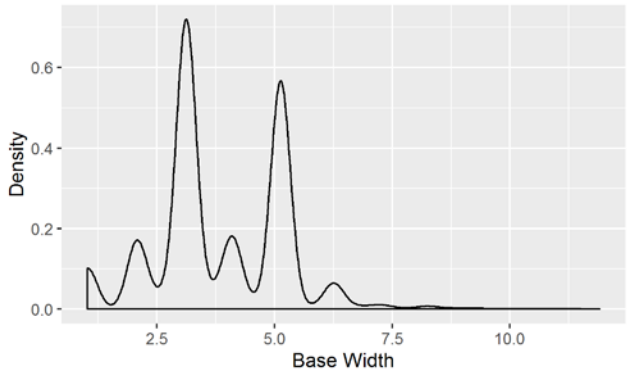


Figure 14. A density plot of the distribution of the Base Width metric in the Formative Evaluation data.

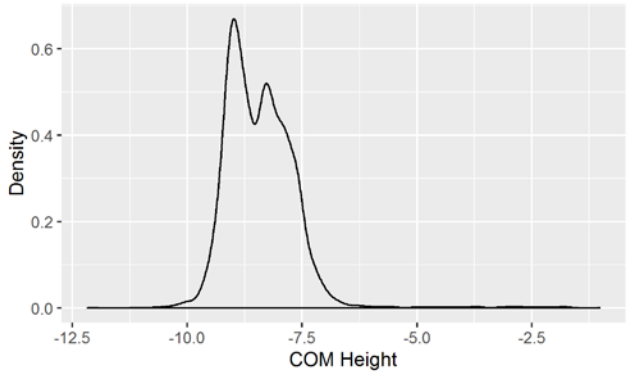


Figure 15. A density plot of the distribution of the COM Height metric in the Formative Evaluation data. Note: COM Height has been reverse-coded, so a higher number on the x-axis is better.

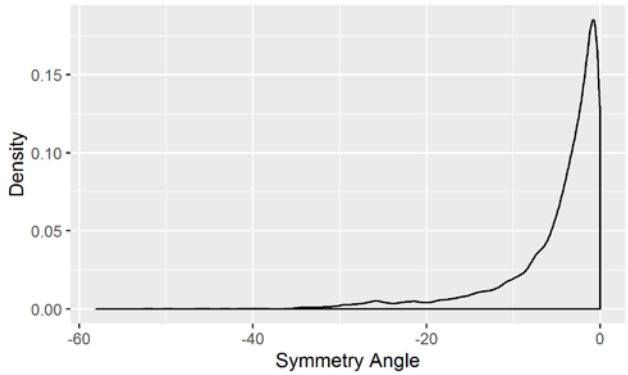


Figure 16. A density plot of the distribution of the Symmetry Angle metric in the Formative Evaluation data. Note: Symmetry Angle has been reverse-coded, so a higher number on the x-axis is better.

Projective Replay Data

The data in this section shows some basic descriptive statistics for the different Projective Replays.

Game Version	Replay Type	Total Attempts	% Successful Attempts
Base	**Retrospective**	4424	82.82
	Literal	4424	77.08
	Flexible	2354	82.97
Glue Blocks	**Literal**	4429	88.64
	Flexible	2140	94.16
No Cliff	**Literal**	4458	75.95
	Flexible	2243	83.75
No Ship	**Literal**	4429	75.16
	Flexible	2907	74.99

Table 18. Descriptive statistics of Projective Replays.

Close-the-Loop Classroom Study

The data in this section describes the new close-the-loop classroom study presented in Chapter 7 . The data in Table 19 is organized by condition and grade.

Condition	Grade	Total Students	Total Attempts	Avg Levels Completed	Std Levels Completed	% All Atmpts Successful	% First Atmpts Successful
Base	**2**	18	655	28.89	12.95	70.99	73.83
	3	22	833	30.09	8.30	72.51	74.62
	Overall	40	1488	29.55	10.51	71.84	74.27
No Ship	**2**	17	661	27.76	9.67	58.85	64.16
	3	20	716	29.25	5.84	65.50	67.49
	Overall	37	1377	28.57	7.76	62.31	65.99
Overall		77	2865	29.08	9.24	67.26	70.37

Table 19. Descriptive statistics of the user population for the close-the-loop study of *RumbleBlocks*.

Figure 17, Figure 18, and Figure 19 show density distributions of the three principle-relevant metrics used in alignment regression. For the close-the-loop study, each plot is further separated by condition. These plots show the raw scores for each metric across all of the data. Note that while these distributions appear very non-normal, this effect is minimized in the regression because of the inclusion of a random intercept for level (see Equation 1).

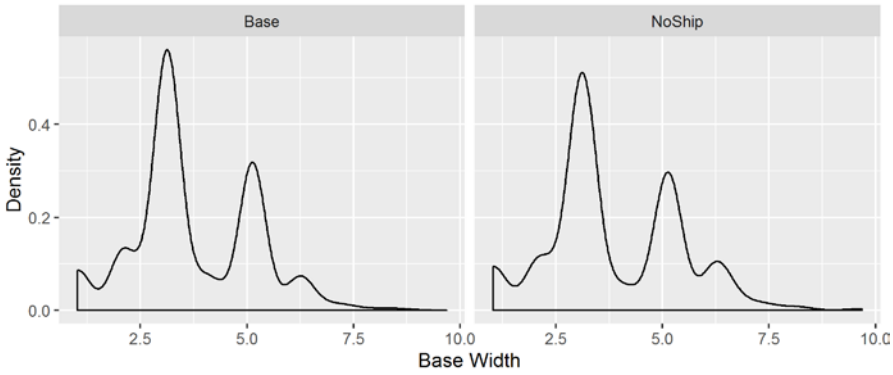


Figure 17. A density plot of the distribution of the Base Width metric in the Close-the-loop data in the Base and No Ship conditions.

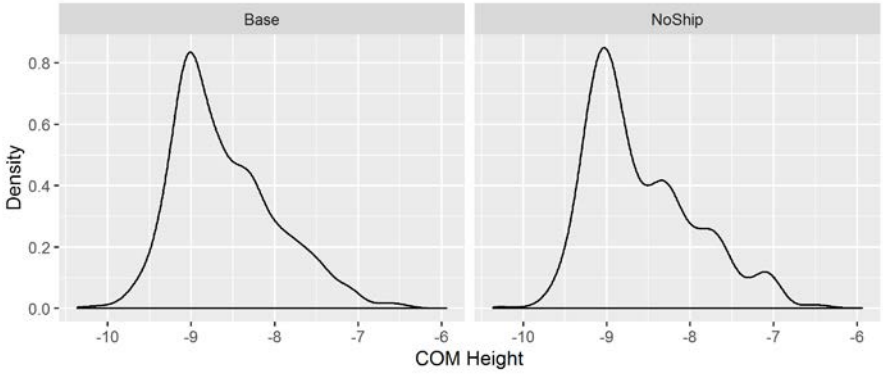


Figure 18. A density plot of the distribution of the COM Height metric in the Close-the-loop data in the Base and No Ship conditions. Note: COM Height has been reverse-coded, so a higher number on the x-axis is better.

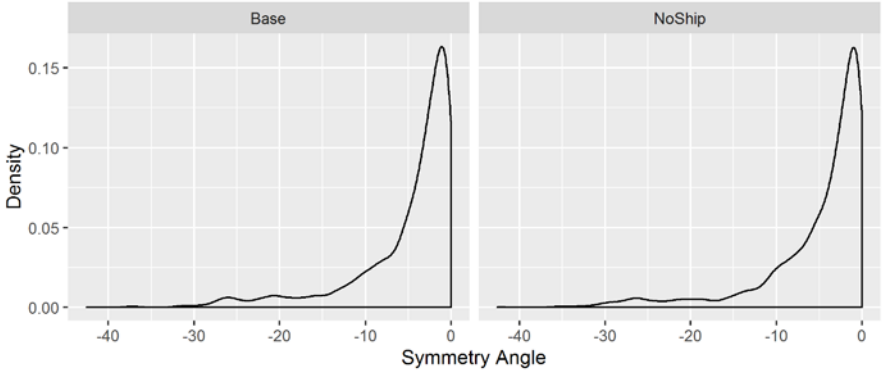


Figure 19. A density plot of the distribution of the Symmetry Angle metric in the Close-the-loop data in the Base and No Ship conditions. Note: Symmetry Angle has been reverse-coded, so a higher number on the x-axis is better.

Appendix B. The TRESTLE Algorithm

Throughout my time as a PhD student, the development of TRESTLE[20] ranks as one of the projects for which I was the proudest to be a part of. That being said, TRESTLE was developed as part of a highly collaborative effort with Christopher MacLellan, and it did not feel right to include it as a contribution in this thesis proper. I do, however, use the algorithm throughout several parts of my research, and I felt it would help readers to have some context on how the algorithm works. I have included a brief description of the algorithm in this appendix for this purpose. We have also previously published a description of TRESTLE in the Advances in Cognitive Systems Journal [93]. A Python implementation is available online at: https://github.com/cmaclell/concept_formation and further documentation can be found at: http://concept-formation.readthedocs.io

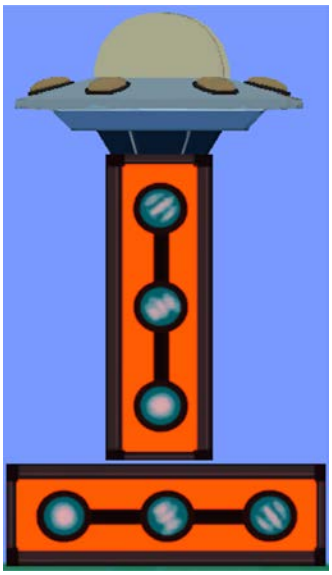
TRESTLE is a model of concept formation based on the COBWEB algorithm [50] that incrementally learns hierarchical concept trees given a set of structured instances. Each concept in a TRESTLE tree is a probabilistic description of the collection of instances stored under it. This probabilistic description is stored in the form of a probability table that tracks how often each attribute-value pair occurs in the underlying instances. Once learned, these concept trees can be used for a number of purposes, such as functioning like a classification system or clustering instances based on their common properties.

One can think of a TRESTLE tree like the classification system of animal taxonomics (i.e., kingdom, phylum, class, order, family, genus, species), but based on a particular set of training data. Alternatively, you could view it like a probabilistic decision tree [169] that optimizes for predicting all attributes of an instance simultaneously rather than predicting a specific target attribute.

Instance Representation

One of the major goals behind the design of TRESTLE that sets it apart from other examples of the COBWEB family of algorithms is being highly permissive in the kinds of data that can be fit into a concept tree. Normally in machine learning, instances must be cast into some kind of flat vector representation in order to be usable. This was one of the reasons we originally developed CFE [61], to make *RumbleBlocks* states amendable to conventional clustering. TRESTLE, on the other hand, is designed to accept instances described by a superset of the common JSON representation (www.json.org). Instances are represented as JSON objects that contain a set of attribute-value pairs. Attributes can be either constant (the standard form), variable (meaning the name is not important), or hidden (meaning they are tracked but not used during categorization). Values can be either nominal (strings), numeric (numbers), or components (nested sub-objects). In addition to these forms, objects can also contain relational attributes that describe relationships between other attributes of the object. Relations are usually used to describe the relationships between sub-objects of a state. An example of a way a *RumbleBlocks* tower would be described to TRESTLE is shown in Figure 20.

[20] TRESTLE is not an acronym. Rendering TRESTLE's name in all caps is a convention we started in its first publication because the other algorithms in the COBWEB family also published their names in all caps, despite not being acronyms themselves. The name TRESTLE was chosen because we intended it to support a bridge between the fields of concept formation and representation learning.



```
{ _Success: “False”,
  ?Block1: {type: “ufo”,
        angle:0.0,
        left:0.1
        right:2.8,
        bottom:4.1,
        top:5.6},
  ?Block2: {type: “rectangle”,
        angle:90.0,
        left:0.9
        right:1.9,
        bottom:1.1,
        top:4.1},
  ?Block3: {type: “rectangle”,
        angle:0.0,
        left:0.0
        right:3.0,
        bottom:0.0,
        top:1.0},
  (On ?Block1 ?Block2): “True”,
  (On ?Block2 ?Block3): “True”}
```

Figure 20. An example of a *RumbleBlocks* tower described as a TRESTLE instance.

In the example in Figure 20, blocks are represented as variable attributes with component values. The Success attribute is hidden (starting with a '_') and has a constant nominal value. Each block's type attribute is a constant nominal value, and its dimensions are constant numeric values. Finally, the *(On …)* relations show examples of relational attributes.

Learning Process

TRESTLE learns from instances incrementally, meaning that its knowledge representation is updated as it encounters new data rather than fit on an entire batch of data at once. This has an additional side effect of making the algorithm susceptible to ordering effects, but in a way that we have previously shown models human ordering effects [93]. The learning process of TRESTLE follows three main phases: partial matching, flattening, and categorization.

Partial Matching

In the partial matching phase, the algorithm tries to match the variable attributes in the new instance to the root concept of the tree. In our instance representation, variable attributes are intended to imply that the name of a given attribute is not important, only that some value exists. This means that it is open to interpretation which other unbound attributes a variable is referring to when it gets integrated into the tree. Partial matching is done by searching over the space of possible bindings for all the variable attributes to find the mapping that produces the greatest number of expected correct guesses of the instance's values. The goal of this step is to align the new instance with the tree's existing knowledge as best as possible before proceeding to later stages. An example mapping is show in Figure 21.

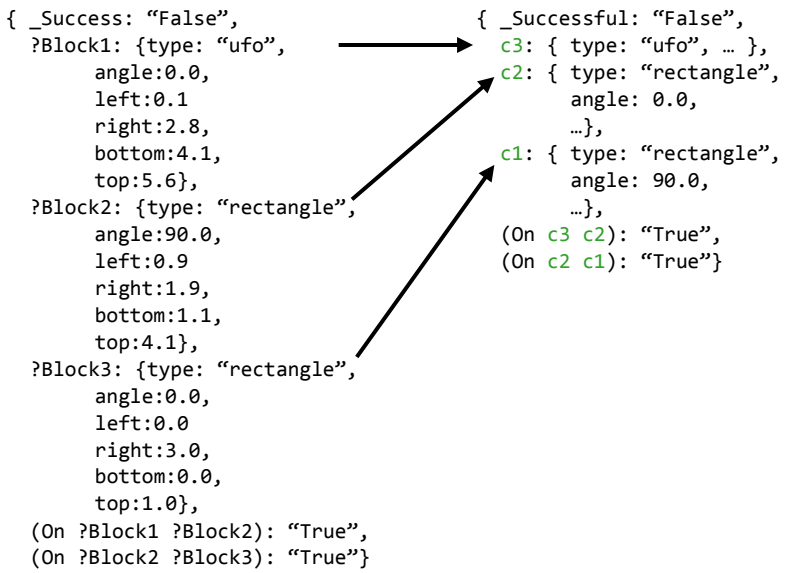


Figure 21. An example partial mapping for an instance in TRESTLE.

In the example mapping in Figure 21, the ufo (?Block1) was renamed to c3 because in more of the previous instances that component happened to be named c3 instead of ?Block1. In this case changing the name would increase the number of expected correct guesses at the root. Note that relations that referenced a variable component have also been changed to maintain the connection.

Flattening

Once the new instance has been partially matched it is converted to a flattened representation. This flat representation removes component values and describes their structure using an equivalent relational syntax.

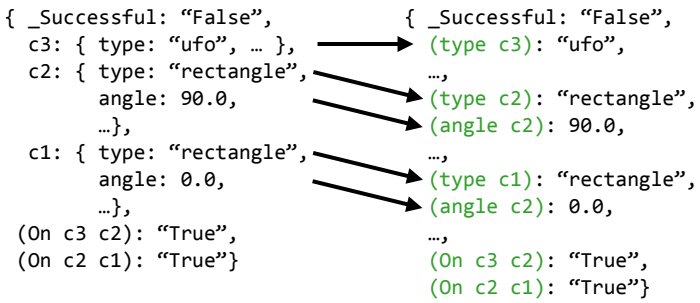


Figure 22. An example of the flattened TRESTLE representation

Categorization

Once the instance has been properly transformed it is integrated into the concept tree. This process follows a method similar to COBWEB [50]. As an instance is incorporated into a concept the frequencies in its probability table are updated to include the attribute-values of the instance. Initially, the instance is added at the root of the tree. Then at each concept it encounters the algorithm considers four possible operations (see Figure 23 for an illustration) to incorporate the instance into its tree: adding the instance to the most similar child concept; creating a new child concept to store the instance; merging the two most similar child concepts and then adding the instance to the resulting concept; and splitting the most similar child concept, promoting its children to be children of the current concept, and recursing. Each operation is simulated and evaluated in terms of its resultant category

utility [50]; whichever operation results in the greatest category utility is applied, and the process continues until the instance is added to a leaf concept or a new child is created.

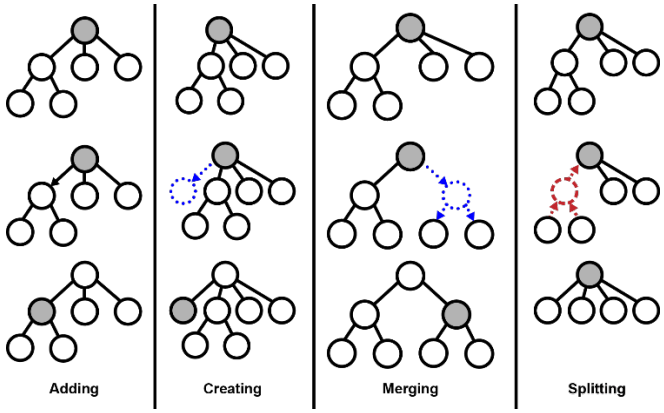


Figure 23. Diagramatic representation of the four COBWEB operations. The grey shaded node represented where the instance is added before, during, and after the operation. Blue dotted lines (creating and merging) represent newly created nodes and red dashed lines (splitting) represent deleted nodes.

Category utility is a measure of the increase in the average number of expected correct guesses of all attribute-values in the children of a node relative to the node itself. The intuition is that, as you move down the tree to more specific descriptions of instances, you are more likely to be correct about what the instance contains. The category utility of a set of children $\{C_1, C_2, \ldots, C_n\}$ is calculated by:

$$\boldsymbol{CU}(\{\boldsymbol{C_1}, \boldsymbol{C_2}, \cdots, \boldsymbol{C_n}\}) = \frac{\sum_{k=1}^{n} \boldsymbol{P}(\boldsymbol{C_k}) \left[\sum_i \sum_j \boldsymbol{P}(\boldsymbol{A_i} = \boldsymbol{V_{ij}} | \boldsymbol{C_k})^2 - \sum_i \sum_j \boldsymbol{P}(\boldsymbol{A_i} = \boldsymbol{V_{ij}})^2\right]}{\boldsymbol{n}}$$

Equation 2. The equation for category utility used in TRESTLE.

Where $P(C_k)$ is the probability of a particular concept given its parent, $P(A_i = V_{ij}|C_k)$ is the probability of attribute A_i having value V_{ij} in the child concept C_k, $P(A_i = V_{ij})$ is the probability of attribute A_i having value V_{ij} in the parent concept, and n is the number of child concepts. Each of these terms can be efficiently computed via a lookup of the probability tables stored in the parent and child concepts.

When dealing with numeric values, TRESTLE uses a normal probability density function to store the probability of different values of a numeric attribute. Under this assumption, the sum of squared attribute-value probabilities in Equation 2 is replaced with an integral of the squared probability density function, which for a normal distribution is the square of the distribution's normalizing constant. Thus when calculating category utility for numeric values the $\sum_i \sum_j P(A_i = V_{ij}|C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2$ in Equation 2 is replaced with $\sum_i \frac{1}{\sigma_{ik}} - \frac{1}{\sigma_i}$ where σ_{ik} is the standard deviation of values for the attribute A_i in child concept C_k and σ_i is the standard deviation of values for the attribute A_i in the parent concept.

Prediction

When TRESTLE is used for prediction or classification of a novel instance, the partial matching and flattening procedures are applied as described above. The categorization process is altered in order to not modify the existing concept hierarchy while predicting. As an instance traverses down the tree the probability tables of the concepts it encounters are not updated. Further, only the creating and adding operations are used at each point. If the creating operation is ever deemed to have the highest category utility then the current concept is returned;

otherwise, the final leaf encountered is returned. Once a concept node is returned, the probability table within that concept can be used to make predictions about the instance's attribute-values either by taking the highest probability value or sampling from the distribution of values as desired.

Clustering

A TRESTLE tree can also be used to create a flat set of clusters of data based on its hierarchical knowledge. When clustering a set of instances, each instance is first categorized by the tree in a non-modifying way similar to the prediction process above. Cluster labels can then be provided by successively splitting concepts, using the normal splitting operation, in the tree and assigning each instance a label of its highest un-split parent, which would initially be the root representing all things.

The number of splits to perform on the tree when clustering is arbitrary, but it can be guided by different model fit statistics such as Akaike Information Criterion (AIC) [2] and Bayesian Information Criterion (BIC) [139] in a similar vein as the x-means clustering algorithm [118]. Both AIC and BIC balance the log-likelihood that the set of clusters would generate their assigned instances against the number of predictors used to generate that model. In the case of a TRESTLE clustering, the number of predictors is the total number of unique-attribute value pairs in the root concept multiplied by the number of clusters. There is no objective way to choose a best heuristic to guide splitting. From our own experience with the algorithm we have generally seen that AIC will lead to more splits resulting in more smaller clusters and BIC will make less splits resulting more commonality between instances.

Appendix C. A Brief Primer on Earthquake Physics

Ever since the early days of ENGAGE, we found it difficult—even as adults—to reason about how exactly earthquakes worked in order to inform our intuitions for designing *RumbleBlocks*. At one point, I was dispatched to have a meeting with Dr. Irving Oppenheim, P.E., a Professor jointly appointed in Carnegie Mellon's departments of Civil and Environmental Engineering, and Architecture. This meeting resulted in a small slide deck of briefing notes (affectionately called "The Math") that I have included as backup slides in all my presentations since as a way of shedding some light on the real complexity of earthquake physics for people who might be confused. I have reproduced the gist of that slide deck here for anyone who might be similarly confused while reading this document. Two of the core intuitions that arise from this deck is that, unintuitive for some, the weight of the tower does not matter, and that one really cannot separate the principles involved from each other, which is what makes alignment in *RumbleBlocks* to difficult.

First, an earthquake is generally modeled as an oscillating process taking place over some duration T, with some number of cycles n and some amplitude D. The earthquake in *RumbleBlocks* has parameters for each of these properties. From these features, we can obtain a frequency $f = n/T$, which can be used to calculate a maximum amount of acceleration applied by the quake as $A = (2\pi f)^2 D$, which can be specified in terms of g's as $\alpha = A/g$ where g is the gravitational constant. Essentially, the goal at this stage is to understand the number of g's of acceleration that the earthquake imparts on the tower.

Next, for the case of a simple tower like the one in Figure 24, the center of mass has a height of h, and is d units from the nearest edge of the tower's base, so in a non-symmetrical tower this would be the smaller of the two distances. Also, the tower has a weight $W = mg$ where m is the mass of the tower and g is the gravitational constant as one would expect. The threshold of motion, i.e. the minimum energy required for the tower to start moving is when $\alpha W > W\ d/h$ or essentially $\alpha > d/h$. The definition of falling (perhaps intuitively) is when the center of mass of an object is no longer over its base. The energy required to cause the tower to topple then is the amount of energy required to lift the center of mass to the point that it is directly above the foot, which is the length of the hypotenuse of the triangle formed with d and h. Thus, resulting in the following equation for energy required:

$$E_r = W\ (\sqrt{h^2 + d^2} - h)$$

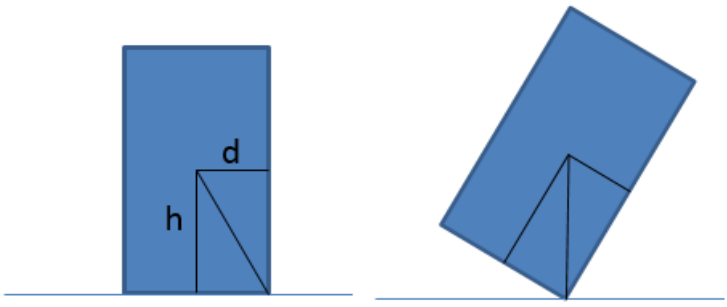


Figure 24. An example tower (left), whose center of mass is at height h and is d units from the nearest edge of the tower's base. And that same tower at the moment of falling (right).

The energy imparted on the tower is the force imparted multiplied by the total tower's weight. The force imparted is ideally αW and the total distance imparted would be $n4D$. Making the total energy imparted on the tower:

$$E_i = \alpha W * n4D$$

However, it is not that simple in reality. Since the earthquake is oscillating, it is not imparting force perfectly on the tower at all times, and not all the motion is contributing to the tower falling. Some examples of this phenomenon are when the tower jumps briefly or when the motion is switching directions and momentarily counteracting its own energy. To account for these issues, you multiply by scaling factors, but these scaling factors are essentially impossible to know *a priori* so they can only be determined empirically. Making the real equation:

$$E_i = X\alpha W * Yn4D$$

where X and Y are some scaling factor between 0 and 1. In our discussion X = 0.25 and Y = 0.10 were offered as possible examples.

Ultimately, a tower falls when the energy imparted is greater than the energy required: $E_i > E_r$ or:

$$X\alpha W * Yn4D > W\,(\sqrt{h^2 + d^2} - h)$$

Note that there is a W (weight) on both sides of the equation that can be factored out, making the final formula:

$$X\alpha * Yn4D > (\sqrt{h^2 + d^2} - h)$$

This final equation means that the weight of the tower will not matter in terms of knocking a tower over with an earthquake. Dr. Oppenheim supported this conclusion and did mention that the weight would matter if it we were instead talking about a projectile impact.

To put the math in terms of the principle-relevant metrics of *RumbleBlocks* consider the abstract example in Figure 25.

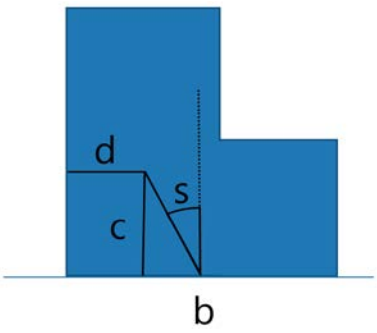


Figure 25. A less regular example tower with a COM Height c, Symmetry Angle s, and Base Width of b.

Given a Base Width b, COM Height c, and a Symmetry Angle s the values of d and h in the original formula can be derived. Simply $h = c$, while $d = \frac{b}{2} - c\tan s$. This results in Energy Required based on PRMS:

$$E_r(PRMs) = \sqrt{c^2 + \left(\frac{b}{2} - c\tan s\right)^2} - c$$

REFERENCES

1. Espen Aarseth. 2007. I Fought the Law: Transgressive Play and The Implied Player. *Proceedings of the 2007 DiGRA International Conference: Situated Play - DiGRA 2007*, 130–133. Retrieved from http://www.digra.org/dl/db/07313.03489.pdf.

2. Hirotogu Akaike. 1973. Information Theory and an Extension of the Maximum Likelihood Principle. *Proceeding of the Second International Symposium on Information Theory*, 267–281.

3. Vincent Aleven. 2010. Rule-Based Cognitive Modeling for Intelligent Tutoring Systems. In *Advances in Intelligent tutoring Systems*. 33–62.

4. Vincent Aleven, Steven Dow, Michael Christel, et al. 2013. Supporting Social-Emotional Development in Collaborative Inquiry Games for K-3 Science Learning. *Proceedings of the 9th Games+Learning+Society Conference - GLS 9.0*.

5. Vincent Aleven and Kenneth R Koedinger. 2013. Knowledge Component Approaches to Learner Modeling. In *Design Recommendations for Intelligent Tutoring Systems: Volume 1 - Learner Modeling*, Robert A Sottilare, Arthur Graesser, Xiangen Hu and Heather Holden (eds.). U.S. Army Resarch Laboratory, 165–182.

6. Vincent Aleven, Bruce M Mclaren, Jonathan Sewall, and Kenneth R Koedinger. 2006. The Cognitive Tutor Authoring Tools (CTAT): Preliminary Evaluation of Efficiency Gains. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems - ITS 2006*, 1–10.

7. Vincent Aleven, Bruce Mclaren, Ido Roll, and Kenneth Koedinger. 2006. Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education* 16, 2: 101–128. http://doi.org/10.1.1.121.9138

8. Vincent Aleven, Eben Myers, Matthew Easterday, and Amy Ogan. 2010. Toward a Framework for the Analysis and Design of Educational Games. *Proceedings of the IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning Toward - DiGITEL 2010*, IEEE, 69–76. http://doi.org/10.1109/DIGITEL.2010.55

9. Christopher Alexander. 1964. *Notes on the Synthesis of Form*. Harvard University Press, Cambridge, MA.

10. Christopher Alexander. 1977. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press.

11. Mike Ambinder. 2014. Making the Best of Imperfect Data: Reflections on an Ideal World. *Keynote to the 1st Annual Symposium on Computer-Human Interaction in Play - CHI PLAY 2014*.

12. Susan A Ambrose, Michael W Bridges, Michele DiPietro, Marsha C Lovett, and Marie K Norman. 2010. *How Learning Works: Seven Research-Based Principles for Smart Teaching*. John Wiley & Sons.

13. Alan Amory. 2006. Game object model version II: a theoretical framework for educational game development. *Educational Technology Research and Development* 55, 1: 51–77. http://doi.org/10.1007/s11423-006-9001-x

14. John R. Anderson, Albert T. Corbett, Kenneth R. Koedinger, and Ray. Pelletier. 1995. Cognitive Tutors: Lessons Learned. *Journal of the Learning Sciences* 4, 2: 167–207. http://doi.org/10.1207/s15327809jls0402_2

15. Lorin W Anderson. 2002. Curricular Alignment: A Re-Examination. *Theory into Practice* 41, 4: 255–260.

16. Leonard A Annetta. 2010. The "I's" have it: A framework for serious educational game design. *Review of General Psychology* 14, 2: 105–112. http://doi.org/10.1037/a0018985

17. Thomas H. Apperley. 2006. Genre and game studies: Toward a critical approach to video game genres. *Simulation and Gaming* 37, 1: 6–23. http://doi.org/10.1177/1046878105282278

18. Ryan S J d Baker, Albert T Corbett, and Angela Z Wagner. 2006. Human Classification of Low-Fidelity Replays of Student Actions. *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, 29–36.

19. Ryan S J D Baker and Kalina Yacef. 2009. The State of Educational Data Mining in 2009 : A Review and Future Visions. *Journal of Educational Data Mining* 1, 1: 3–16. http://doi.org/http://doi.ieeecomputersociety.org/10.1109/ASE.2003.1240314

20. Sasha Barab, Michael Thomas, Tyler Dodge, Robert Carteaux, and Hakan Tuzun. 2005. Making Learning Fun: Quest Atlantis, a Game without Guns. *Educational Technology Research and Development* 53, 1: 86–107.

21. Tiffany Barnes. 2005. The Q-matrix Method: Mining Student Response Data for Knowledge. *Proceedings of the 20th National Conference on Artificial Intelligence*.

22. Kent Beck, Mike Beedle, Arie van Bennekum, et al. 2001. Manifesto for Agile Software Development.

23. Marion Boberg, Evangelos Karapanos, Jussi Holopainen, and Andrés Lucero. 2015. PLEXQ: Towards a Playful Experiences Questionnaire. *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY '15*, 381–391. http://doi.org/10.1145/2793107.2793124

24. Richard Buchanan. 1996. Wicked Problems in Design Thinking. In *The Idea of Design*, Victor Margolin and Richard Buchanan (eds.). MIT Press, Cambridge, MA, 3–20.

25. Marion Buchenau and Jane Fulton Suri. 2000. Experience prototyping. *Proceedings of the conference on Designing interactive systems processes, practices, methods, and techniques - DIS '00*, ACM Press, 424–433. http://doi.org/10.1145/347642.347802

26. Brian Burg, Richard Bailey, Andrew J. Ko, and Michael D. Ernst. 2013. Interactive record/replay for web application debugging. *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*: 473–484. http://doi.org/10.1145/2501988.2502050

27. Stuart K Card, Thomas P Moran, and Allen Newell. 1983. The Human Information-Processor. In *The Psychology of Human-Computer Interaction*. 23–97.

28. Hao Cen, Kenneth Koedinger, and Brian Junker. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems - ITS 2006*, Springer, 164–175.

29. Catherine C Chase, Erik Harpstead, and Vincent Aleven. Inciting transfer of learning beyond the game: adapting contrast-based instruction for educational games. *In preperation*.

30. Judeth Oden Choi, Jodi Forlizzi, Michael Christel, Rachel Moeller, Mackenzie Bates, and Jessica Hammer. 2016. Playtesting with a Purpose. *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY '16*, ACM Press, 254–265. http://doi.org/10.1145/2967934.2968103

31. Michael G Christel, Scott Stevens, Matt Champer, et al. 2013. Beanstalk : A Unity Game Addressing Balance Principles, Socio-Emotional Learning and Scientific Inquiry. *Proceedings of the International Games Innovation Conference*, IEEE, 36–39.

32. Michael G Christel, Scott M Stevens, Bryan S Maher, et al. 2012. RumbleBlocks: Teaching science concepts to young children through a unity game. *Proc CGAMES 2012*, IEEE, 162–166. http://doi.org/10.1109/CGames.2012.6314570

33. Douglas B. Clark, Emily E. Tanner-Smith, and Stephen S. Killingsworth. 2016. Digital Games, Design, and Learning: A Systematic Review and Meta-Analysis. *Review of Educational Research* 86, 1: 79–122. http://doi.org/10.3102/0034654315582065

34. Richard E Clark. 1994. Media will never influence learning. *Educational Technology Research and Development* 42, 2: 21–29. http://doi.org/10.1007/BF02299088

35. Doug Clow. 2012. The learning analytics cycle: Closing the loop effectively. *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge - LAK '12*, ACM Press, 134. http://doi.org/10.1145/2330601.2330636

36. S Alan Cohen. 1987. Instructional Alignment: Searching for a Magic Bullet. *Educational Researcher* 16, 8: 16–20.

37. Thomas M Connolly, Elizabeth A Boyle, Ewan MacArthur, Thomas Hainey, and James M Boyle. 2012. A systematic literature review of empirical evidence on computer games and serious games. *Computers & Education* 59, 2: 661–686. http://doi.org/10.1016/j.compedu.2012.03.004

38. Seth Cooper, Firas Khatib, Adrien Treuille, et al. 2010. Predicting protein structures with a multiplayer online game. *Nature* 466, 7307: 756–60. http://doi.org/10.1038/nature09304

39. Greg Costikyan. 2002. I Have No Words & I Must Design: Toward a Critical Vocabulary for Games. *Computer Games and Digital Cultures*, 9–33.

40. Sabrina Culyba. Transformational Games: A Field Guide for Design Leaders. In Preparation.

41. Susan L. Davis-Becker and Chad W. Buckendahl. 2013. A proposed framework for evaluating alignment studies. *Educational Measurement: Issues and Practice* 32, 1: 23–33. http://doi.org/10.1111/emip.12002

42. Girlie C Delacruz, Gregory K W K Chung, and Eva L Baker. 2010. *Validity Evidence for Games as Assessment Environments (CRESST Report 773)*. Los Angeles, CA.

43. Alena Denisova, A Imran Nordin, and Paul Cairns. 2016. The Convergence of Player Experience Questionnaires. *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY '16*, ACM Press, 33–37. http://doi.org/10.1145/2967934.2968095

44. Patrick Dickinson. 2001. Instant Replay: Building a Game Engine with Reproducible Behavior. *Gamasutra*. Retrieved from http://www.gamasutra.com/view/feature/131466/instant_replay_building_a_game_.php

45. Mary Jo Dondlinger. 2007. Educational Video Game Design: A Review of the Literature. *Journal of Applied Educational Technology* 4, 1: 21–31.

46. Anders Drachen and Alessandro Canossa. 2009. Towards gameplay analysis via gameplay metrics. *Proc. MindTrek 2009*, ACM Press, 202. http://doi.org/10.1145/1621841.1621878

47. Anders Drachen, Magy Seif El-Nasr, and Alessandro Canossa. 2013. Game Analytics - The Basics. In *Game Analytics*, Magy Seif El-Nasr, Anders Drachen and Alessandro Canossa (eds.). Springer London, London, 13–40. http://doi.org/10.1007/978-1-4471-4769-5

48. Hugh Dubberly, Shelley Evenson, and Rick Robinson. 2008. The analysis-systhesis bridge model. *Interactions* 15, 2: 57–61. http://doi.org/10.1145/1340961.1340976

49. Olive Jean Dunn. 2014. Multiple Comparisons Among Means. *Journal of the American Statistical Association* 56, 293: 52–64.

50. Douglas H Fisher. 1987. Knowledge Acquisition Via Incremental Conceptual Clustering Learning ~ Element Performance Element. *Machine Learning* 2: 139–172.

51. Tracy Fullerton. 2014. *Game Design Workshop*. CRC Press, Boca Raton, FL.

52. James Paul Gee. 2003. *What video games have to teach us about learning and literacy*. Palgrave Macmillan, New York.

53. Chaim Gingold and Chris Hecker. 2006. Advanced Prototyping. *Game Developers Conference*. Retrieved from http://chrishecker.com/Advanced_prototyping

54. C Girard, Jean Ecalle, and Annie Magnan. 2012. Serious games as new educational tools: how effective are they? A meta-analysis of recent studies. *Journal of Computer Assisted Learning* 29, 3: 207–219. http://doi.org/10.1111/j.1365-2729.2012.00489.x

55. Jamie Griesemer. Design in Detail: Changing the Time Between Shots for the Sniper Rifle from 0.5 to 0.7 Seconds for Halo 3. *Presented at the 22nd Game Developer's Conference - GDC 2010*.

56. M. P. Jacob Habgood and Shaaron E. Ainsworth. 2011. Motivating Children to Learn Effectively: Exploring the Value of Intrinsic Integration in Educational Games. *Journal of the Learning Sciences* 20, 2: 169–206. http://doi.org/10.1080/10508406.2010.508029

57. Greg Hamerly and Charles Elkan. 2003. Learning the k in k-means. *Proc. NIPS '03*.

58. Erik Harpstead and Vincent Aleven. 2015. Using Empirical Learning Curve Analysis to Inform Design in an Educational Game. *Proceedings of the 2nd ACM SIGCHI annual symposium on Computer-human interaction in play - CHI PLAY 2015*, ACM Press, 197–207. http://doi.org/10.1145/2793107.2793128

59. Erik Harpstead, Christopher J. MacLellan, Vincent Aleven, and Brad A Myers. 2014. Using extracted features to inform alignment-driven design ideas in an educational game. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, ACM Press, 3329–3338. http://doi.org/10.1145/2556288.2557393

60. Erik Harpstead, Christopher J. MacLellan, Vincent Aleven, and Brad A Myers. 2015. Replay Analysis in Open-Ended Educational Games. In *Serious Games Analytics*, Christian Sebastian Loh, Yanyan Sheng and Dirk Ifenthaler (eds.). Springer International Publishing, Cham, 381–399. http://doi.org/10.1007/978-3-319-05834-4_17

61. Erik Harpstead, Christopher J Maclellan, Kenneth R Koedinger, Vincent Aleven, Steven P Dow, and Brad A Myers. 2013. Investigating the Solution Space of an Open-Ended Educational Game Using Conceptual Feature Extraction. *Prococeedings of the 6th International Conference on Educational Data Mining - EDM 2013*, 51–58.

62. Erik Harpstead, Brad A Myers, and Vincent Aleven. 2013. In search of learning: facilitating data analysis in educational games. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, ACM Press, 79. http://doi.org/10.1145/2470654.2470667

63. Marc Hassenzahl. 2013. User Experience and Experience Design. January 2011: 1–15.

64. Marc Hassenzahl and Noam Tractinsky. 2006. User experience - a research agenda. *Behaviour & Information Technology* 25, 2: 91–97. http://doi.org/10.1080/01449290500330331

65. Stephanie Heintz and Effie Lai-chong Law. 2015. The Game Genre Map: A Revised Game Classification. *Proceedings of the 2nd ACM SIGCHI annual symposium on Computer-human interaction in play - CHI PLAY 2015*, 175–184. http://doi.org/10.1145/2793107.2793123

66. David R Hill, Seth A King, Christopher J Lemons, and Jane N Partanen. 2012. Fidelity of Implementation and Instructional Alignment in Response to Intervention Research. *Learning Disabilities Research & Practice (Wiley-Blackwell)* 27, 3: 116–124. http://doi.org/10.1111/j.1540-5826.2012.00357.x

67. Christoffer Holmgård, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. 2016. Evolving models of player decision making: Personas versus clones. *Entertainment Computing* 16: 95–104. http://doi.org/10.1016/j.entcom.2015.09.002

68. Johan Huizinga. 1950. *Homo Ludens: A Study of the Play-element in Culture*. Beacon Press, Boston, MA.

69. Robin Hunicke, Marc Leblanc, and Robert Zubek. 2004. MDA: A Formal Approach to Game Design and Game Research. *Proc. of the AAAI Workshop on Challenges in Game AI*, 1–5.

70. Alexander Jaffe, Alex Miller, Erik Andersen, and Ye Liu. 2012. Evaluating Competitive Game Balance with Restricted Play. *Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment - AIIDE 2012*, 26–31. Retrieved from http://www.aaai.org/ocs/index.php/AIIDE/AIIDE12/paper/viewFile/5470/5692

71. Bonnie E. John and David E. Kieras. 1996. The GOMS family of user interface analysis techniques: comparison and contrast. *ACM Transactions on Computer-Human Interaction* 3, 4: 320–351. http://doi.org/10.1145/235833.236054

72. Yasmin B Kafai and Deborah A Fields. 2009. Cheating in virtual worlds: Transgressive designs for learning. *On the Horizon* 17, 1: 12–20. http://doi.org/10.1108/10748120910936117

73. Rilla Khaled, Mark J Nelson, and Pippin Barr. 2013. Design Metaphors for Procedural Content Generation in Games. *Proceedings of the 31st International ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2013*, ACM Press, 1509–1518.

74. Jun H Kim, Daniel V Gunn, Eric Schuh, Bruce Phillips, Randy J Pagulayan, and Dennis Wixon. 2008. Tracking real-time user experience (TRUE): A comprehensive instrumentation solution for complex systems. *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, ACM Press, 443. http://doi.org/10.1145/1357054.1357126

75. Kenneth R Koedinger, Ryan S J d Baker, Kyle Cunningham, Alida Skogsholm, Brett Leber, and John Stamper. 2010. A Data Repository for the EDM community: The PSLC DataShop. In *Handbook of Educational Data Mining*, Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy and Ryan S.J.d. Baker (eds.). 43–55.

76. Kenneth R Koedinger, Julie L Booth, and David Klahr. 2013. Instructional Complexity and the Science to Constrain It. *Science* 342, 6161: 935–937.

77. Kenneth R Koedinger, Albert T Corbett, and Charles Perfetti. 2012. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science* 36, 5: 757–798. http://doi.org/10.1111/j.1551-6709.2012.01245.x

78. Kenneth R Koedinger and Mitchell J. Nathan. 2004. The Real Story Behind Story Problems: Effects of Representations on Quantitative Reasoning. *The Journal of the Learning Sciences* 13, 2: 129–164.

79. Kenneth R Koedinger, John C Stamper, Elizabeth A Mclaughlin, and Tristan Nixon. 2013. Using Data-Driven Discovery of Better Student Models to Improve Student Learning. *Proc. AIED 2013*, 421–430.

80. Ron Kohavi, Alex Deng, Brian Frasca, Roger Longbotham, Toby Walker, and Ya Xu. 2012. Trustworthy Online Controlled Experiments: Five Puzzling Outcomes Explained. *Proc. KDD 2012*, ACM Press, 786–794.

81. Robert B. Kozma. 1994. Will media influence learning? Reframing the debate. *Educational Technology Research and Development* 42, 2: 7–19. http://doi.org/10.1007/BF02299087

82. David R Krathwohl. 2002. A Revision of Bloom's Taxonomy: An Overview. *Theory into Practice* 41, 4: 212–218.

83. Pat Langley and Tom M Mitchell. 1982. Learning from Solution Paths: An Approach to the Credit Assignment Problem. *AI Magazine* 3, 2: 48–52.

84. Jill H Larkin. 1989. Display-Based Problem Solving. In *Complex Information Processing: The Impact of Herbert A. Simon*. Psychology Press, 319–341.

85. Jacqueline P Leighton and Rebecca J Gokiert. 2005. The Cognitive Effects of Test Item Features: Informing Item Generation by Identifying Construct Irrelevant Variance. *Proc. NCME 2005*, 1–26.

86. Nan Li. 2013. Integrating Representation Learning and Skill Learning in a Human-Like Intelligent Agent.

87. Conor Linehan, George Bellord, Ben Kirman, Zachary H Morford, and Bryan Roche. 2014. Learning curves: Analysing Pace and Challenge in Four Successful Puzzle Games. *Proceedings of the 1st ACM*

SIGCHI annual symposium on Computer-human interaction in play - CHI PLAY 2014, ACM Press, 181–190. http://doi.org/10.1145/2658537.2658695

88. Christian Sebastian Loh and Yanyan Sheng. 2015. Measuring Expert Performance for Serious Games Analytics: From Data to Insights. In *Serious Games Analytics*. Springer International Publishing, Cham, 101–134. http://doi.org/10.1007/978-3-319-05834-4_5

89. Christian Sebastian Loh, Yanyan Sheng, and Dirk Ifenthaler (eds.). 2015. *Serious Games Analytics*. Springer International Publishing, Cham. http://doi.org/10.1007/978-3-319-05834-4

90. Collin Lynch, Kevin D Ashley, Niels Pinkwart, and Vincent Aleven. 2009. Concepts , Structures , and Goals : Redefining Ill-Definedness. *International Journal of Artificial Intelligence in Education* 19: 253–266.

91. David J C MacKay. 2003. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press. Retrieved from http://www.inference.org.uk/mackay/itila/book.html

92. Christopher J Maclellan. 2017. Computational Models of Human Learning: Applications for Tutor Development, Behavior Prediction, and Theory Testing.

93. Christopher J. MacLellan, Erik Harpstead, Vincent Aleven, and Kenneth R. Koedinger. 2016. TRESTLE: A Model of Concept Formation in Structured Domains. *Advances in Cognitive Systems* 4: 131–150.

94. Christopher J Maclellan, Erik Harpstead, Rony Patel, and Kenneth R Koedinger. 2016. The Apprentice Learner Architecture: Closing the loop between learning theory and educational data. *Proceedings of the 9th International Conference on Educational Data Mining - EDM '16*, 151–158.

95. Christopher J Maclellan, Eliane Stampfer Wiese, Noboru Matsuda, and Kenneth R Koedinger. 2014. SimStudent : Authoring Expert Models by Tutoring. *Proceedings of the 2nd Annual GIFT Users Symposium - GIFTSym2*.

96. Andrea Martone and Stephen G Sireci. 2009. Evaluating Alignment Between Curriculum, Assessment, and Instruction. *Review of Educational Research* 79, 4: 1332–1361. http://doi.org/10.3102/0034654309341375

97. Noboru Matsuda, William W Cohen, and Kenneth R Koedinger. 2014. Teaching the Teacher: Tutoring SimStudent Leads to More Effective Cognitive Tutor Authoring. *International Journal of Artificial Intelligence in Education* 25, 1. http://doi.org/10.1007/s40593-014-0020-1

98. Michael C Medlock, Dennis Wixon, Mark Terrano, Ramon L Romero, and Bill Fulton. 2002. Using the RITE method to improve products ; a definition and a case study. *Proceedings of the Usability Professionals Association - UPA 2002*. http://doi.org/10.1.1.116.6486

99. Gabriel Menotti. 2014. Videorec as gameplay: Recording playthroughs and video game engagement. *The Italian Journal of Game Studies*, 3: 81–92.

100. Pejman Mirza-Babaei, Lennart E Nacke, John Gregory, Nick Collins, and Geraldine Fitzpatrick. 2013. How does it play better? exploring user testing and biometric storyboards in games user research. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, ACM Press, 1499. http://doi.org/10.1145/2470654.2466200

101. Robert J Mislevy, Russell G Almond, and Janice F Lukas. 2003. A Brief Introduction to Evidence-centered Design. July.

102. Robert J Mislevy, Andreas Oranje, Malcolm I Bauer, et al. 2014. Pyschometrica Considerations in Game-Based Assessments. 160. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0

103. Tom M Mitchell. 1997. *Machine Learning*. McGraw-Hill, Boston, MA.

104. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540: 529–533. http://doi.org/10.1038/nature14236

105. Jonathan Moizer and Jonathan Lean. 2010. Toward Endemic Deployment of Educational Simulation Games: A Review of Progress and Future Recommendations. *Simulation & Gaming* 41, 1: 116–131. http://doi.org/10.1177/1046878109359052

106. Glenford J Myers, Corey Sandler, and Tom Badgett. 2011. *The Art of Software Testing*. Wiley.

107. Lennart Nacke and Anders Drachen. 2011. Towards a Framework of Player Experience Research. *Proc. EPEX 2011*.

108. Lennart E Nacke and Craig A Lindley. 2008. Flow and Immersion in First-person Shooters: Measuring the Player's Gameplay Experience. *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*: 81–88. http://doi.org/10.1145/1496984.1496998

109. Alfredo Nantes, Ross Brown, and Frederic Maire. 2008. A Framework for the Semi-Automatic Testing of Video Games. *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference*, 197–202.

110. National Research Council. 2012. *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. The National Academies Press.

111. Alan S Neal and Roger M Simons. 1984. Playback: A method for evaluating the usability of software and its documentation. *IBM Systems Journal* 23, 1: 82–96. http://doi.org/10.1147/sj.231.0082

112. Allen Newell and Herbert A Simon. 1972. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ.

113. A. Imran Nordin, Alena Denisova, and Paul Cairns. 2014. Too Many Questionnaires: Measuring Player Experience Whilst Playing Digital Games. *Seventh York Doctoral Symposium on Computer Science & Electronics*, October: 69–75.

114. Jaclyn Ocumpaugh, Ryan S J d Baker, and Ma Mercedes T Rodrigo. 2012. Baker-Rodrigo Observation Method Protocol (BROMP) 1.0 Training Manual version 1.0.

115. Jennifer K Olsen and Vincent Aleven. 2017. Statistically Modeling Individual Students' Learning over Successive Collaborative Practice Opportunities. 54, 1: 123–138.

116. V. Elizabeth Owen and Richard Halverson. 2013. ADAGE (Assessment Data Aggregator for Game Environments): A Click-Stream Data Framework for Assessment of Learning in Play. *Proceedings of the 9th Games+Learning+Society Conference - GLS 9.0*, ETC Press, 248–254.

117. V Elizabeth Owen, Dennis Ramirez, Allison Salmon, and Richard Halverson. 2014. Capturing Learner Trajectories in Educational Games through ADAGE (Assessment Data Aggregator for Game

Environments): A Click-Stream Data Framework for Assessment of Learning in Play. *American Educational Research Association Annual Meeting*: 1–7.

118. Dan Pelleg and Andrew Moore. 2000. X-means: Extending K-means with efficient estimation of the number of clusters. *Proceedings of the Seventeenth International Conference on Machine Learning table of contents*: 727–734. http://doi.org/10.1007/3-540-44491-2_3

119. Jan L Plass, Bruce D Homer, Charles Kinzer, Jonathan M Frye, and Ken Perlin. 2011. Learning mechanics and assessment mechanics for games for learning. Retrieved from http://createx.alt.ed.nyu.edu/classes/2505/reading/Plass et al LAMechanics 2505.pdf

120. Morgan S. Polikoff and Gavin W. Fulmer. 2013. Refining Methods for Estimating Critical Values for an Alignment Index. *Journal of Research on Educational Effectiveness* 6, 4: 380–395. http://doi.org/10.1080/19345747.2012.755593

121. Morgan S. Polikoff and Andrew C. Porter. 2014. Instructional Alignment as a Measure of Teaching Quality. *Educational Evaluation and Policy Analysis* 20, X: 1–18. http://doi.org/10.3102/0162373714531851

122. Andrew C. Porter. 2002. Measuring the Content of Instruction: Uses in Research and Practice. *Educational Researcher* 31, 7: 3–14. http://doi.org/10.3102/0013189X031007003

123. Andrew C Porter and John L Smithson. 2002. Alignment of Assessments, Standards, and Instruction Using Curriculum Indicator Data. 1–9. Retrieved from papers3://publication/uuid/B03EED83-4364-4023-BDB6-4D5E0A306F88

124. William M Rand. 1971. Objective Criteria for the Evaluation of Clustering Methods. *Journal of American Statistical Association* 66, 336: 846–850.

125. Don Rawitsch. 2017. Classic Game Postmortem: "Oregon Trail." *Presented at the Game Developers Conference - GDC 2017*. Retrieved from http://www.gdcvault.com/play/1024251/Classic-Game-Postmortem-Oregon-Trail

126. Steven Ritter and Kenneth R Koedinger. 1996. An Architecture For Plug-in Tutor Agents. *Journal of Artificial Intelligence in Education* 7, 3/4: 315–347.

127. Elizabeth Rowe, Jodi Asbell-Clarke, and Ryan S Baker. 2015. Serious Games Analytics to Measure Implicit Science Learning. In *Serious Games Analytics*. Springer International Publishing, Cham, 343–360. http://doi.org/10.1007/978-3-319-05834-4_15

128. Elizabeth Rowe, Ryan S J d Baker, Jodi Asbell-clarke, Emily Kasman, and William J Hawkins. 2014. Building Automated Detectors of Gameplay Strategies to Measure Implicit Science Learning. *Proceedings of the International Conference on Educational Data Mining - EDM '14*, 337–338.

129. Jonathan P Rowe, Bradford W Mott, Scott W McQuiggan, Jennifer L Robison, Sunyoung Lee, and James C Lester. 2009. CRYSTAL ISLAND: A Narrative-Centered Learning Environment for Eighth Grade Microbiology. *Proceedings of the Workshop on Intelligent Educational Games at AIED 2009*, 11–20. Retrieved from http://people.ict.usc.edu/~lane/AIED2009-IEG-WorkshopProceedings-FINAL.pdf#page=19

130. André a. Rupp, Matthew Gushta, Robert J. Mislevy, and David Williamson Shaffer. 2010. Evidence-centered design of epistemic games: Measurement principles for complex learning environments. *The Journal of Technology Learning and Assessment* 8, 4: 3–41.

131. Stuart J Russell and Peter Norvig. 2009. *AI a Modern Approach*. Pearson.

132. J Sabourin, B Mott, and Lester. J. 2011. Modeling Learner Affect with Theoreticallly Grounded Dynamic Bayesian Networks. *Proceedings of the Fourth International Conference on Affective Computing and Intelligent Interaction*, 286–295. http://doi.org/10.1007/978-3-642-24600-5_32

133. Katie Salen and Eric Zimmerman. 2004. *Rules of Play*. MIT Press, Cambridge, MA.

134. Tom Schaul. 2013. A video game description language for model-based or interactive learning. *IEEE Conference on Computatonal Intelligence and Games, CIG*. http://doi.org/10.1109/CIG.2013.6633610

135. Jesse Schell. 2008. *The Art of Game Design: A Book of Lenses*. Morgan Kaufmann, Burlington, MA.

136. Donald A. Schön. 1982. *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York, New York, USA.

137. Donald A. Schön. 1992. Designing as reflective conversation with the materials of a design situation. *Research in Engineering Design* 3, 3: 131–147. http://doi.org/10.1007/BF01580516

138. Donald A. Schön and G. Wiggins. 1992. Kinds of Seeing and their Function in Designing. *Design Studies* 19: 135–156.

139. Gideon Schwarz. 1978. Estimating the Dimension of a Model. *The Annals of Statistics* 6, 2: 461–464.

140. Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa (eds.). 2013. *Game Analytics: Maximizing the Value of Player Data*. Springer.

141. Howard J Seltman. 2015. *Experimental Design and Analysis*.

142. Mohammad Shaker, Noor Shaker, and Julian Togelius. 2010. Evolving Playable Content for Cut the Rope through a Simulation-Based Approach. *Proceedings of the Ninth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment Evolving - AIIDE 2010*, 72–78.

143. Brett E. Shelton, Tom Satwicz, and Tom Caswell. 2011. Historical Perspectives on Games and Education from the Learning Sciences. *International Journal of Game-Based Learning* 1, 3: 83–106. http://doi.org/10.4018/ijgbl.2011070106

144. Valerie J. Shute, Matthew Ventura, and Yoon Jeon Kim. 2013. Assessment and Learning of Qualitative Physics in Newton's Playground. *The Journal of Educational Research* 106, 6: 423–430. http://doi.org/10.1080/00220671.2013.832970

145. Valerie J Shute and Lubin Wang. 2015. Measuring Problem Solving Skills in Portal 2. *E-learning Eystems, Environments and Approaches: Theory and implementation*: 11–25. http://doi.org/10.1007/978-3-319-05825-2

146. Stefan Slater, Srećko Joksimović, Vitomir Kovanovic, Ryan S Baker, and Dragan Gasevic. 2016. Tools for Educational Data Mining: A Review. *Journal of Educational and Behavioral Statistics* XX, X: 1–22. http://doi.org/10.3102/1076998616666808

147. Adam M. Smith, Erik Andersen, Michael Mateas, and Zoran Popović. 2012. A case study of expressively constrainable level design automation tools for a puzzle game. *Proceedings of the 7th International Conference on Foundations of Digital Games - FDG '12*, ACM Press, 156. http://doi.org/10.1145/2282338.2282370

148. Adam M Smith, Eric Butler, and Zoran Popovi. 2013. Quantifying over Play: Constraining Undesirable Solutions in Puzzle Design. *Proc. FDG 2013*, 221–228.

149. Adam M Smith and Michael Mateas. 2011. Computational caricatures: Probing the game design process with ai. *Artificial Intelligence in the Game Design Process - Papers from the 2011 AIIDE Workshop*: 19–24. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-84877984109&partnerID=40&md5=13865cf22d1aa43c1457b0413fdd5b98

150. Adam M Smith, Mark J Nelson, and Michael Mateas. 2009. Computational Support for Play Testing Game Sketches. *Proceedings of the Fifth Artificial Intelligence for Interactive Digital Entertainment Conference - AIIDE 2009*, 167–172. Retrieved from http://www.aaai.org/ocs/index.php/AIIDE/AIIDE09/paper/viewFile/690/1061

151. Brian A Smith and Shree K Nayar. 2016. Mining Controller Inputs to Understand Gameplay. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16*, ACM Press, 157–168. http://doi.org/10.1145/2984511.2984543

152. Kurt Squire. 2008. Open-Ended Video Games: A Model for Developing Learning for the Interactive Age. In *The Ecology of Games: Connecting Youth, Games, and Learning*, Katie Salen (ed.). MIT Press, Cambridge, MA, 167–198. http://doi.org/10.1162/dmal.9780262693646.167

153. John C Stamper and Kenneth R Koedinger. 2011. Human-machine student model discovery and improvement using Data. *Proceedings of the 15th International Conference on Artificial Intelligence in Education - AIED '11*, Springer, 353–360.

154. Alexandra To, Elaine Fath, Eda Zhang, et al. 2016. Tandem Transformational Game Design: A Game Design Process Case Study. *Proceedings of the International Academic Conference on Meaningful Play*.

155. Maryam Tohidi, William Buxton, Ronald Baecker, and Abigail Sellen. 2006. Getting the Right Design and the Design Right : Testing Many Is Better Than One. 1243–1252.

156. Alan M Turing. 1950. Computing Machinery and Intelligence. *Mind* 59, 236: 433–460.

157. Anders Tychsen. 2008. Crafting User Experience via Game Metrics Analysis. *Proc. NordiCHI 2008*, 20–22.

158. Jason Vandenberghe. 2012. The 4fs of Game Design. *Game Developer*, 44–46.

159. Kurt VanLehn. 2006. The Behavior of Tutoring Systems. *Int. J. Artif. Intell. Ed.* 16, 3: 227–265. Retrieved from http://dl.acm.org/citation.cfm?id=1435351.1435353

160. Kurt Vanlehn, Kenneth R Koedinger, Alida Skogsholm, et al. 2007. What's in a Step? Toward General, Abstract Representations of Tutoring System Log Data. *Proceedings of the 11th International Conference on User Modeling - UM 2007*, Springer Berlin Heidelberg, 455–459.

161. Kurt VanLehn, Stellan Ohlsson, and Rod Nason. 1994. Applications of Simulated Students: An Exploration. *Journal of Artificial Intelligence in Education* 5, 2: 1–42.

162. Günter Wallner. 2015. Sequential Analysis of Player Behavior. *CHI PLAY '15 Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, 349–358. http://doi.org/10.1145/2793107.2793112

163. Matthijs J Warrens. 2008. On the Equivalence of Cohen's Kappa and the Huber-Arabie Adjusted Rand Index. *Journal of Classification* 25: 177–183.

164. Ben G. Weber and Michael Mateas. 2009. A data mining approach to strategy prediction. *2009 IEEE Symposium on Computational Intelligence and Games*, Ieee, 140–147. http://doi.org/10.1109/CIG.2009.5286483

165. Ben Weber, M Mateas, and a Jhala. 2012. Learning from Demonstration for Goal-Driven Autonomy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2007: 1176–1182. Retrieved from http://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/viewPDFInterstitial/5090/5537

166. Grant Wiggins and Jay McTighe. 2005. *Understanding by Design*. Pearson.

167. Jason R. Wilson, Kenneth D. Forbus, and Matthew D. McLure. 2013. Am I really scared? A multi-phase computational model of emotions. *Proceedings of the Second Annual Conference on Advances in Cognitive Systems*, 289–304.

168. Brian M Winn. 2009. The Design, Play, and Experience Framework. In *Handbook of Research on Effective Electronic Gaming in Education*. IGI Global, 1010–1024. http://doi.org/10.4018/978-1-59904-808-6.ch058

169. Xindong Wu, Vipin Kumar, J. Ross Quinlan, et al. 2007. *Top 10 algorithms in data mining*. http://doi.org/10.1007/s10115-007-0114-2

170. Georgios N Yannakakis and Julian Togelius. 2015. A Panorama of Artificial and Computational Intelligence in Games. *IEEE Transaction on Computational Intelligence and AI in Games* 7, 4: 317–335.

171. Michael F. Young, Stephen Slota, Andrew B. Cutter, et al. 2012. Our Princess Is in Another Castle: A Review of Trends in Serious Gaming for Education. *Review of Educational Research* 82, 1: 61–89. http://doi.org/10.3102/0034654312436980

172. Alexander Zook, Brent Harrison, and Mark O Riedl. 2015. Monte-Carlo Tree Search for Simulation-based Strategy Analysis. *Proceedings of the 10th Conference on the Foundations of Digital Games*.